VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

ADVOKATIS - systém pro správu advokátní kanceláře

ADVOKATIS - IS for Law Companies

2015 Martin Porostlý

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Martin Porostlý**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: ADVOKATIS - systém pro správu advokátní kanceláře
ADVOKATIS - IS for Law Companies

Zásady pro vypracování:

Nynější systémy pro správu advokátních kanceláří jsou sice velmi obsáhlé a funkční, ale o to více jim chybí jednoduché uživatelské rozhraní a podpora mobilních zařízení. Úkolem studenta bude vytvořit informační systém, který bude přehledný a bude podporovat mobilní zařízení. Systém bude umožňovat advokátům správu jejich kanceláří, jako např. evidovat klienty, organizovat schůzky či přidělovat jednotlivým uživatelům úkoly.

Úkoly:
1. Prostudujte existující IS pro podporu advokátních kanceláří a zhodnoťte jejich vlastnosti a nedostatky.
2. Navrhněte a naimplementujte informační systém, který bude obsahovat:
a) Webový server přístupný jak z místní sítě, tak z internetu.
b) Responzivní design, tak aby byl správně zobrazitelný na mobilních telefonech, tabletech a počítačích.
c) Plně funkční webovou aplikaci.
d) Třídu pro komunikaci s veřejnými registry, které používají advokáti.
3. Porovnejte IS s existujícími systémy a zhodnoťte jeho přínosy.

Seznam doporučené odborné literatury:

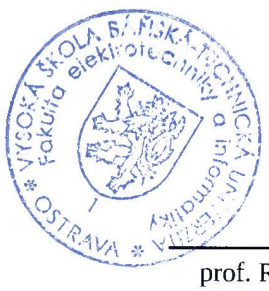
Podle pokynů vedoucího bakalářské práce.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Peter Chovanec**

Datum zadání: 01.09.2014

Datum odevzdání: 07.05.2015



doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 5. května 2015

.....

Chtěl bych zde poděkovat vedoucímu této bakalářské práce Ing. Petrovi Chovancovi za odborné vedení, za pomoc a rady při tvorbě informačního systému ADVOKATIS. Dále bych rád poděkoval JUDr. Janě Mikulové a celé její advokátní kanceláři za konzutace, připomínky a průběžné testování mojí aplikace.

Abstrakt

Tato bakalářská práce se zabývá návrhem, implementací a testováním webového informačního systému určeného pro menší advokátní kanceláře. Cílem je vytvořit přehlednou, moderní a mobilní aplikaci, která bude umožňovat advokátům správu jejich kanceláří, jako např. evidovat klienty, organizovat schůzky či přidělovat jednotlivým uživatelům úkoly. V práci je popsán celý proces vývoje IS, počínaje prozkoumáním již existujících systémů pro správu advokátní kanceláře, analýzou požadavků, výběrem použitých technologií, implementací IS a jeho otestováním.

Klíčová slova: advokátní kancelář, advokát, právo, spis, kontakt, dokument, informační systém

Abstract

This bachelor thesis describes the design, implementation and testing of web information system for smaller law companies. The goal is implementation of well-arranged, modern and mobile application that will allow lawyers to manage their companies, e.g. to register clients, to organize meetings, or to adding tasks to individual users. This bachelor thesis contains entire process of IS development, a survey of currently existing systems for law companies, analysis of customer requirements, description of choosen technologies, implementation of IS and its testing.

Keywords: law company, lawyer, law, document, contact, file, information system

Seznam použitých zkratek a symbolů

ACL	–	Access Control List
ADVOKATIS	–	Informační systém pro správu advokátní kanceláře
AJAX	–	Asynchronous JavaScript and XML
CSRF	–	Cross-Site Request Forgery
CSS	–	Cascading Style Sheets
HTML	–	Hyper Text Markup Language
HTTP	–	Hyper Text Transfer Protocol
IS	–	Informační systém
JSON	–	JavaScript Object Notation
MVC	–	Model-View-Controller
MySQL	–	My Structured Query Language
PHP	–	PHP Hyper Text Preprocessor

Obsah

Seznam obrázků

Seznam výpisů zdrojového kódu

1 Úvod

V dnešním světě je běžné, že stále více lidí potřebuje k vyřešení svých problémů právní pomoc. Často slýcháme slova jako rozvod, dědictví, žaloba nebo exekuce, kdy je třeba se obrátit na právníka v dané oblasti. Vzniká tak řada velkých i malých právnických firem, které nabízejí své právní služby. Aby tyto služby byly dostatečně kvalitní, je nutné, aby jejich kanceláře byly také dobře vedeny a organizovány.

Momentálně existují dva IS pro správu advokátní kanceláře: Jurisoft a AdvocatusDigital.

Webový IS Jurisoft [8] poskytuje všechny běžné agendy, jako je evidence spisů, klientů, termínů a lhůt, sledování insolvencí, vedení služebních cest, fakturaci atd. Ačkoliv je tato aplikace poměrně obsáhlá, působí velmi nepřehledně hlavně díky tabulkám, které je potřeba horizontálně posouvat i při vysokém rozlišení displeje.

Webový IS AdvocatusDigital [9] působí po grafické stránce moderněji než IS Jurisoft a využívá mnoho moderních HTML, CSS a JavaScriptových knihoven. Aplikace umožňuje evidovat spisy, klienty, dokumenty, úkoly a lze ji napojit na email. Podstatným nedostatkem je chybějící možnost vést jakékoliv finance ve spise a je tedy vhodná pouze pro správu základních agend s použitím dodatečného programu na účtování. Ovládaní této aplikace je nelogické a uživatelsky nepřívětivé. Aplikace neumožňuje komunikovat s veřejnými rejstříky, např. pro automatické stáhnutí datové schránky kontaktu. V aplikaci se mi líbilo jednotné vyhledávací pole, které vyhledává jak ve spisech, tak v klientech zároveň.

Z vlastní zkušenosti vím, že stále existuje mnoho menších právnických firem, které nepoužívají žádný systém pro organizaci svých případů, a právě těmto firmám bych rád pomohl. Navíc dnešní uspěchaná doba vyžaduje, aby advokát byl schopen reagovat na požadavky klientů i mimo sídlo své kanceláře. Obě předchozí aplikace je možné používat pouze vzdáleně, kdy je IS spuštěn na webovém serveru a uživatelé se mohou připojit pouze přes internet. Advokátní kancelář je tedy závislá na svém poskytovateli internetu. Ani jedna z aplikací není přizpůsobena pro použití na mobilním zařízení. Z těchto důvodů jsem vytvořil IS, který jsem pojmenoval ADVOKATIS.

Cílem práce bylo tedy vytvořit informační systém tak, aby jeho uživateli byly převážně menší advokátní kanceláře. Z tohoto důvodu jsem se snažil, aby byla aplikace přehledná, jednoduchá a disponovala funkcemi, které advokátovi co nejvíce zjednoduší práci se svými spisy a klienty. Příkladem může být komunikace s veřejnými rejstříky, kdy se po vyplnění identifikačního čísla kontaktu automaticky stáhnou jeho detailní informace a číslo datové schránky. Aplikaci ADVOKATIS jsem navrhl tak, aby fungovala na jakémkoliv počítači s přístupem na internet. Advokát by se tak mohl podívat do svých spisů z domova, nebo třeba v průběhu služební cesty. Pokud je advokát ve své kanceláři, připojuje se k systému pomocí lokální sítě, ale pokud je mimo svou kancelář, tak přes internet. Při implementaci této aplikace jsem se zaměřil také na uživatele, kteří by aplikaci ADVOKATIS chtěli využít i na jiných zařízeních, než je počítač, např. mobilní telefon nebo tablet. Proto jsem využil responzivního vzhledu, který zaručuje, že je aplikace zcela funkční na jakémkoliv zařízení. Tato poměrně nová metoda je dnes téměř standardem

při implementaci webové aplikace.

Při vývoji tohoto informačního systému jsem spolupracoval s několika advokátními kancelářemi, zejména pak s Advokátní kanceláří JUDr. Jany Mikulové z Ostravy.

2 Analýza a návrh

Hlavním úkolem této kapitoly je popsat procesy, které nastanou, když klient poprvé navštíví advokátní kancelář. Následně budou popsány jednotlivé role a práva pracovníků kanceláře a definovány funkční a nefunkční požadavky informačního systému ADVOKATIS.

2.1 Procesy

1. Nový klient se osobně dostaví do kanceláře, případně si po telefonu sjedná schůzku.
2. Poté co klient seznámí advokáta se svým právním problémem, může nastat následující situace:
   (a) Jednorázová ústní konzultace, kdy se o této konzultaci evidence nevede.
   (b) Sepsání smlouvy, žaloby atd.
   (c) Právní zastupování advokátem.
3. V prvních dvou případech se advokát s klientem předem ústně domluví na odměně za právní službu. V případě (c) je klient poučen o množství úkonů, které budou v případě zastoupení potřeba a dále se advokát s klientem dohodne o případné výši smluvní odměny za vyřešení celého případu, či na odměně za jeden právní úkon. V případě, že se klient s advokátem nedohodnou smluvně, je klient poučen o výši mimosmluvní odměny, která je dána dle advokátního tarifu (vyhláška č. 177/1996 Sb., o odměnách advokátů a náhradách advokátů v platném znění). Advokát také poučí klienta o množství úkonů a vyžádá si zálohu za právní zastoupení.

   Typy odměn při právním zastupování:

   Mimosmluvní: Jednotlivé úkony jsou pevně oceněny různou částkou dle advokátního tarifu.

   Smluvní: Tento typ odměny může být buď pevný, kdy se advokát s klientem dopředu domluví na ceně za vyřešení celého případu, nebo procentuální, kdy se domluví na konkrétním procentu odměny z žalované částky. Odměna může být také hodinová, což znamená, že se advokát s klientem dohodne na ceně za každou započatou hodinu, kdy advokát pracuje na případu.
4. Klient je vždy dopředu seznámen s typem odměny.
5. Klientovi je fyzicky založen spis a vyznačí se datum, kdy advokát klienta začal zastupovat. Dále do spisu zapíše, zda-li se jedná o odměnu smluvní, či mimosmluvní. Do spisu také advokát zapisuje veškeré provedené úkony jako je např. odeslání pošty, termíny schůzek, přijaté zálohy, či jiné advokátovy výdaje (kolky, cestovné atd.).
6. Jakmile je případ ukončen, vystaví advokát klientovi fakturu.

2.2 Datová analýza

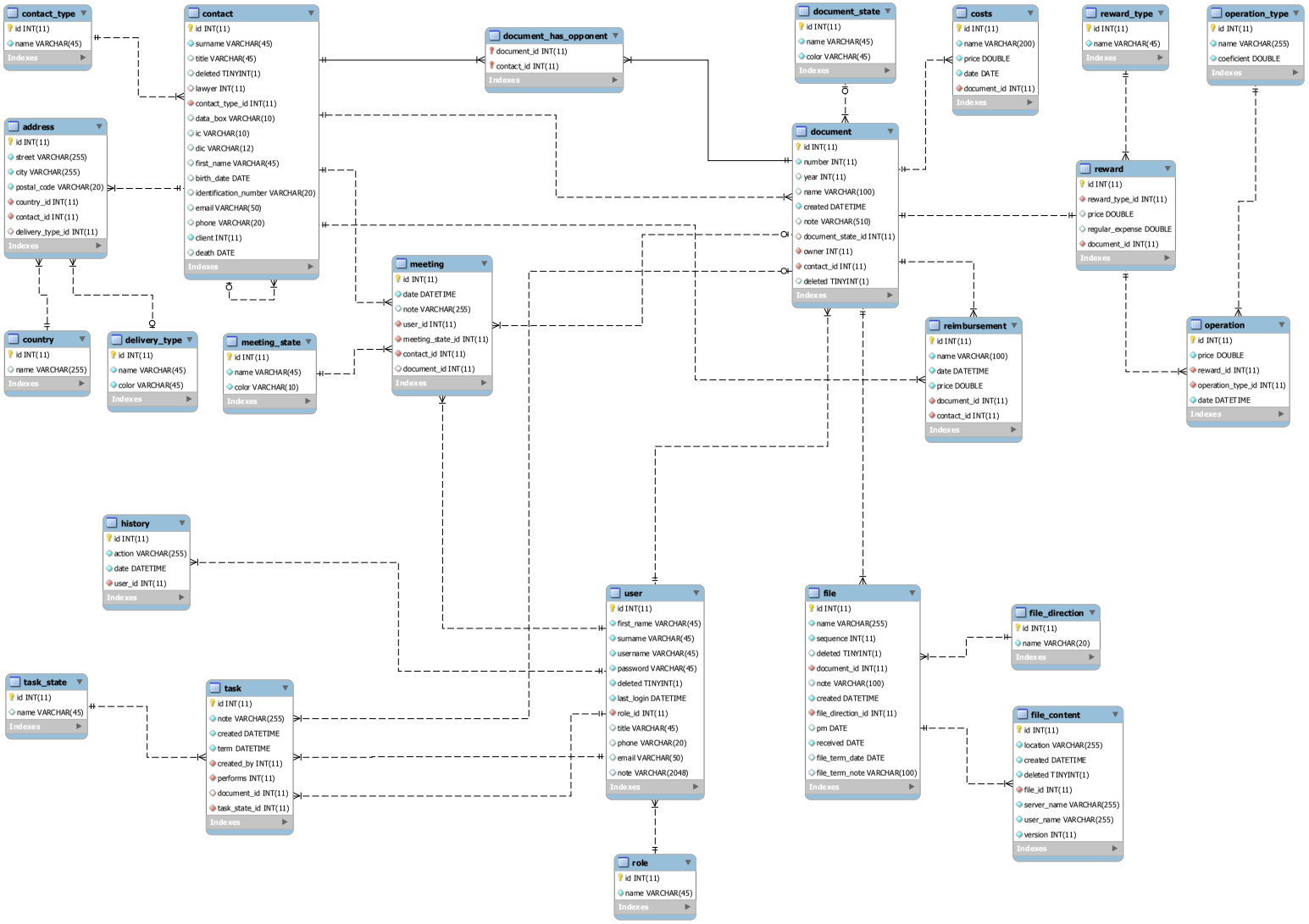
2.2.1 Vstupy do systému

Jedná se o informace, které IS ADVOKATIS příjme, zpracuje a uloží do databáze. V následujícím seznamu lze vidět jednotlivé vstupy do systému a jejich popisy.

- **Spis**
  vstupní data: *číslo, rok, název, datum a čas vytvoření, poznámka, stav spisu, vlastník, klient, protistrany*
- **Kontakt**
  vstupní data: *titul, jméno, příjmení, právní zástupce, typ osoby, typ vystupování ve spisech, datum narození/založení právnické osoby, datum úmrtí/zániku právnické osoby, rodné číslo, identifikační číslo, daňové identifikační číslo, datová schránka, email, telefon*
- **Uživatel**
  vstupní data: *titul, jméno, příjmení, přezdívka, heslo, role, email, telefon, poznámka*
- **Odměna**
  vstupní data: *typ odměny, cena, hotové výdaje, spis*
- **Advokátní úkon**
  vstupní data: *cena, odměna, typ úkonu, datum a čas vytvoření*
- **Adresa**
  vstupní data: *ulice a č.p., město, PSČ, země, kontakt, typ adresy*
- **Schůzka**
  vstupní data: *datum a čas, poznámka, kontakt, uživatel, stav schůzky, spis*
- **Úkol**
  vstupní data: *poznámka, datum a čas vytvoření, termín, autor, vykonavatel, spis, stav úkolu*
- **Dokument**
  vstupní data: *název, datum a čas vytvoření, datum a čas přijetí, datum právní moci, spis, poznámka, směr dokumentu, termín, poznámka k termínu, příloha*
- **Náklad**
  vstupní data: *popis, cena, datum vložení, spis*
- **Platba**
  vstupní data: *popis, cena, datum vložení, spis, plátce*
- **Stav spisu, stav schůzky, typ adresy**
  vstupní data: *název, barva*

2.2.2 Datový model

Na základě předchozí kapitoly byl navrhnut datový model informačního systému (viz. obrázek č. 1). Popis jednotlivých tabulek a datový slovník lze nalézt v přílohách A a B.



Obrázek 1: ER diagram systému ADVOKATIS

2.3 Uživatelské role

Tato aplikace by se neobešla bez rozlišování typů uživatelů, už jen z toho důvodu, že by nebylo zřetelné, zda se daný úkol týká sekretářky, nebo advokáta. Proto jsou v aplikaci definovány následující role:

1. **Advokát**
   Právník, který vykonal advokátní zkoušky, je zapsán v seznamu advokátů České advokátní komory a klientům za úplatu řeší právní problémy. Má veškerá práva v systému. Může tedy vytvářet i mazat všechny uživatele, pracovat se spisy, s kontakty, se schůzkami, s úkoly a má přístup do nastavení systému.

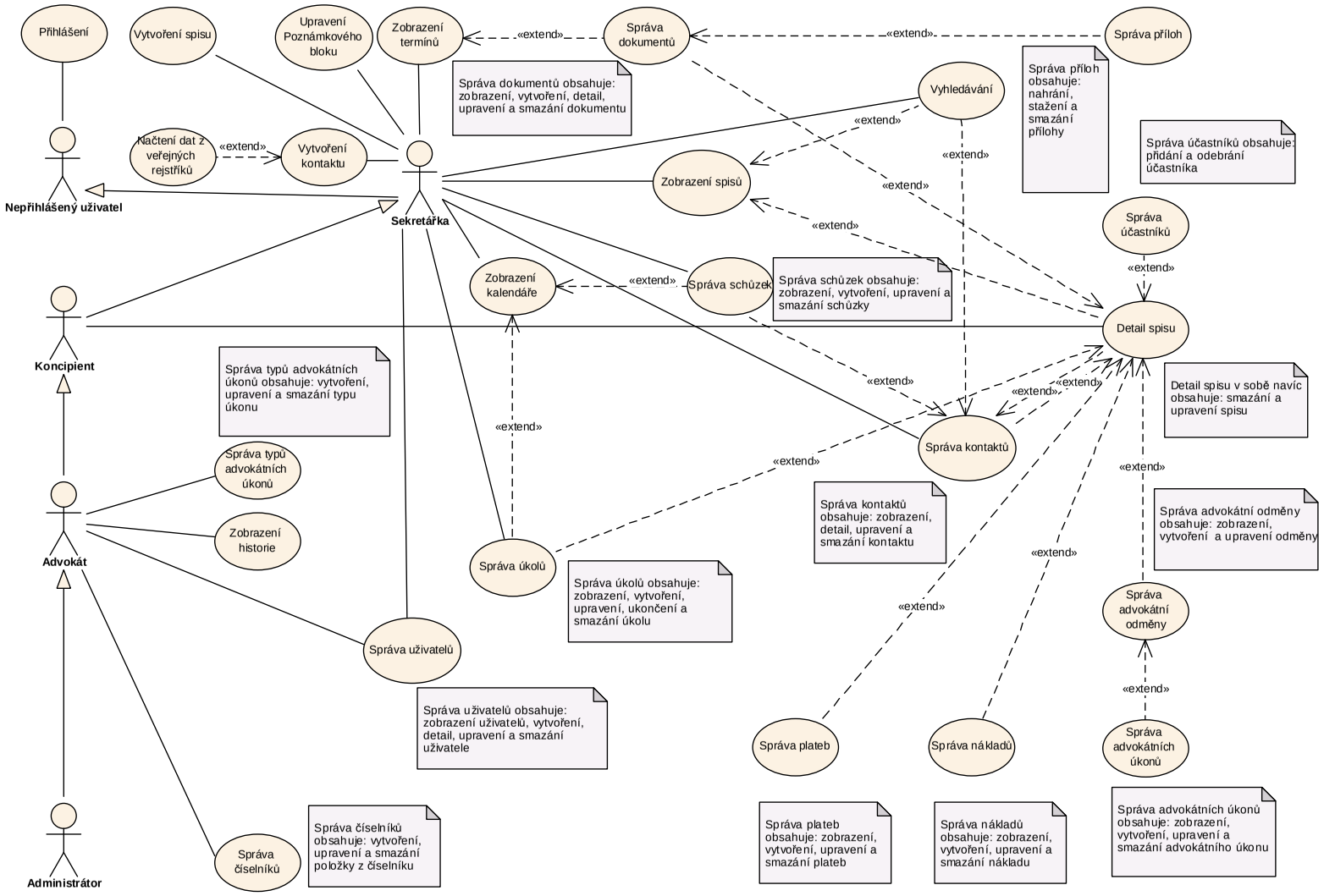
2. **Sekretářka**
   Uživatel, který se primárně stará o organizaci kanceláře. Může pracovat se spisy, s kontakty, se schůzkami, s úkoly a upravovat pouze své osobní informace. Nemůže vést svůj vlastní spis.

3. **Koncipient**
   Jedná se o zaměstnance advokáta, který většinou vykonává advokátní praxi. Může pracovat se spisy, s kontakty, se schůzkami a s úkoly a upravovat pouze své osobní informace. Nemůže vést svůj vlastní spis, stejně jako sekretářka.

4. **Administrátor**
   Osoba, která má na starost chod aplikace. Kromě nemožnosti vést svůj spis má veškerá práva, stejně jako advokát.

Všichni uživatelé (sekretářka, koncipient, advokát, administrátor) mají velmi podobné práva, a to z toho důvodu, že v menších advokátních kancelářích se často stává, že je advokát např. u soudu, ale kancelář musí stále fungovat i za jeho nepřítomnosti.

2.4 Use case diagram



Obrázek 2: Use case diagram

2.5 Funkční analýza

Popis jednotlivých funkcí systému podle kategorií, ve kterých se logicky nacházejí.

Správa spisů
Tabulky: document, document_state, document_has_opponent
Zodpovědnost: Advokát, Koncipient, Sekretářka, Administrátor

1. **Vytvoření spisu:** Při zakládání nového spisu je potřeba vyplnit základní spisové údaje, připojit patřičného klienta, připojit vedoucího spisu, kterým musí být vždy advokát, a připojit jednu nebo více protistran.
2. **Vyhledání spisu:** Lze vyhledat konkrétní spis, či vícero spisů zároveň. Dále je možné vyhledávat spisy podle roků.
3. **Zobrazení všech spisů:** Zobrazí seznam spisů včetně jejich stavu, vedoucích a dalších informací.
4. **Zobrazení detailu spisu:** Zobrazí detail spisu, ve kterém lze vidět základní spisové informace, nahrané dokumenty, připojené kontakty, odměnu advokáta, náklady, přijaté platby a úkoly týkající se spisu.
5. **Úprava základních údajů o spise:** Lze upravovat základní informace o spise. Taktéž je možno do spisu vkládat, upravovat a mazat dokumenty, kontakty, odměnu advokáta, náklady, přijaté platby a úkoly ve spise.
6. **Smazání spisu:**
   Zodpovědnost: Advokát, Administrátor
   Smazanému spisu je nastaven příznak `deleted` tak, aby v případě potřeby mohl být znovu z databáze obnoven.

Správa dokumentů
Tabulky: file, file_content, file_direction
Zodpovědnost: Advokát, Koncipient, Sekretářka, Administrátor
Každý spis může obsahovat více dokumentů. Ke konkrétnímu dokumentu lze vytvořit i více příloh (verzování příloh), ale primární bude vždy ta, která byla nahrána jako poslední.

1. **Vytvoření dokumentu:** Pro vytvoření dokumentu je potřeba vyplnit jeho název a směr (odchozí nebo příchozí). Nepovinně lze vyplnit poznámku, datum přijetí dokumentu nebo datum právní moci. Taktéž je k dokumentu možno nahrát přílohu.
2. **Zobrazení dokumentů ve spise:** Zobrazí seznam všech dokumentů, které patří k vybranému spisu.
3. **Smazání dokumentu:** Smazanému dokumentu a všem jeho přílohám je nastaven příznak `deleted` a dále už se ve spise nezobrazují.

4. **Zobrazení detailu dokumentu:** Zobrazí podrobný detail dokumentu včetně všech jeho nahraných příloh. Jednotlivé přílohy lze stáhnout nebo smazat.

5. **Úprava informací o dokumentu:** Umožňuje upravit parametry dokumentu, jako např. název, poznámka, datum přijetí, směr, datum právní moci, termín nebo poznámku k termínu. Taktéž je možno k dokumentu nahrát další přílohu (novou verzi přílohy).

Správa kontaktů
Tabulky: contact, contact_type
Zodpovědnost: Advokát, Koncipient, Sekretářka, Administrátor

1. **Vytvoření kontaktu:** Pro jeho vytvoření je potřeba zadat základní informace jako je jméno, adresa, rodné číslo atd. Pokud se jedná o firmu je potřeba vyplnit IČ a datovou schránku. Při vytváření je možné využít automatického propojení s rejstříky ARES [1] a PVS [2] pro načtení informací o kontaktu podle jeho IČ. Tuto funkci zobrazuje diagram aktivit v obrázku č. 3.

2. **Zobrazení všech kontaktů:** Zobrazí seznam všech kontaktů.

3. **Zobrazení detailu kontaktu:** V detailu kontaktu mohou uživatelé vidět základní informace o kontaktu, seznam spisů, ve kterých kontakt vystupuje, právního zástupce kontaktu (pouze v případě, že je kontakt veden jako protistrana) a seznam schůzek s tímto kontaktem.

4. **Úprava kontaktu:** Lze upravovat základní informace o kontaktu, jako např. jméno, datum narození, číslo datové schránky, poznámku atd.

5. **Smazání kontaktu:** Kontaktu je nastaven příznak `deleted` tak, aby mohl být v budoucnu případně obnoven.

6. **Vyhledání kontaktu:** Vyhledávat kontakt lze podle nejrůznějších kritérií jako je část jména, RČ, IČ, nebo data narození.

Správa schůzek
Tabulky: meeting, meeting_state
Zodpovědnost: Advokát, Koncipient, Sekretářka, Administrátor

1. **Vytvoření schůzky:** Pro vytvoření schůzky je potřeba vyplnit její datum a čas, který uživatel se jí zůčastní, kterého kontaktu se týká a její popis. Nepovinně lze schůzku přiřadit ke spisu.

2. **Zobrazení všech schůzek:** Zobrazí seznam všech schůzek.

[1] Administrativní registr ekonomických subjektů [30]
[2] Portál veřejné správy [31]

3. **Zobrazení svých schůzek:** Zobrazí seznam schůzek jen vybraného uživatele.
4. **Smazání schůzky:** Permanentně smaže schůzku z databáze.
5. **Úprava schůzky:** Umožňuje upravit parametry schůzky, jako např. její datum, uživatele, přiřazený kontakt, poznámku nebo případně spis, ke kterému je přiřazena.

Správa úkolů
Tabulky: task, task_state
Zodpovědnost: Advokát, Koncipient, Sekretářka, Administrátor

1. **Vytvoření úkolu:** Pro vytvoření úkolu je potřeba vyplnit jeho název, termín a uživatele, který úkol vykonává. Nepovinně lze úkol přiřadit ke spisu.
2. **Zobrazení všech úkolů:** Zobrazí seznam všech úkolů.
3. **Zobrazení svých úkolů:** Zobrazí seznam úkolů, které vykonává jen vybraný uživatel.
4. **Smazání úkolu:** Permanentně smaže úkol z databáze.
5. **Úprava úkolu:** Umožňuje upravit parametry úkolu, jako např. název, termín nebo uživatele, který jej vykonává.
6. **Ukončení úkolu:** Pouze uživatel, který úkol vykonává jej může ukončit.

Správa kalendáře
Tabulky: meeting, meeting_state, task, task_state, file
Zodopovědnost: Advokát, Koncipient, Sekretářka, Administrátor

1. **Zobrazení kalendáře:** Zobrazí všechny schůzky, úkoly a termíny pro konkrétní den.
2. **Vytvoření události v kalendáři:** Umožňuje vytvořit nový úkol nebo schůzku s datem, které je aktuálně vybráno v kalendáři.

Správa uživatelů
Tabulky: user, role
Zodpovědnost: Advokát, Administrátor

1. **Vytvoření uživatele:**
   Pro vytvoření uživatele je potřeba zadat základní informace, jako je jméno, příjmení, přezdívku, heslo a vybrat roli. Nepovinně je možno vyplnit titul, telefon a email.

2. **Úprava uživatele:**
   Zodpovědnost: Advokát, Koncipient, Sekretářka, Administrátor
   Všechny role mohou upravit informace o svém účtu. Role advokát a administrátor mohou upravit informace o jakémkoliv uživateli. Dále si uživatelé mohou změnit své heslo. Role advokát a administrátor mohou měnit hesla všem uživatelům.

3. **Zobrazení detailu uživatele:**
   Zodpovědnost: Advokát, Koncipient, Sekretářka, Administrátor
   Zobrazí veškeré informace o uživateli, jako např. jméno, příjmení, titul, roli, datum posledního přihlášení atd.

4. **Zobrazení všech uživatelů:** Zobrazí seznam všech uživatelů.

5. **Smazání uživatele:** Uživateli je přidělen příznak `deleted` tak, aby mohl být v budoucnu případně obnoven.

Správa číselníků
Tabulky: meeting_state, task_state, document_state
Zodpovědnost: Advokát, Administrátor
Číselníky jsou následující: Typ adresy, Úkony v odměnách, Stav spisu

1. **Zobrazení všech číselníků:** Zobrazí číselníky: stav spisu, stav schůzky a typ adresy. Číselníky jsou zobrazeny včetně všech položek, které obsahují.

2. **Vložení dat do číselníku:** Do číselníků stav spisu a typ adresy lze přidat novou položku. Je potřeba vyplnit její jméno a barvu.

3. **Smazání dat v číselníku:** Položky v kterémkoliv číselníku lze permanentně smazat z databáze.

4. **Úprava dat v číselníku:** U položek v číselnících stav spisu a typ adresy lze upravit název a barvu. U položek v číselníku stav schůzky lze upravit pouze barvu.

Správa typů advokáních úkonů
Tabulky: meeting_state, task_state, document_state
Zodpovědnost: Advokát, Administrátor

1. **Zobrazení všech typů advokátních úkonů:** Zobrazí všechny typy advokátních úkonů včetně jejich názvu a koeficientu, který je potřebný pro výpočet odměny advokáta.

2. **Vytvoření nového typu advokátního úkonu:** Pro vytvoření nového typu advokátního úkonu je potřeba vyplnit jeho název a koeficient.

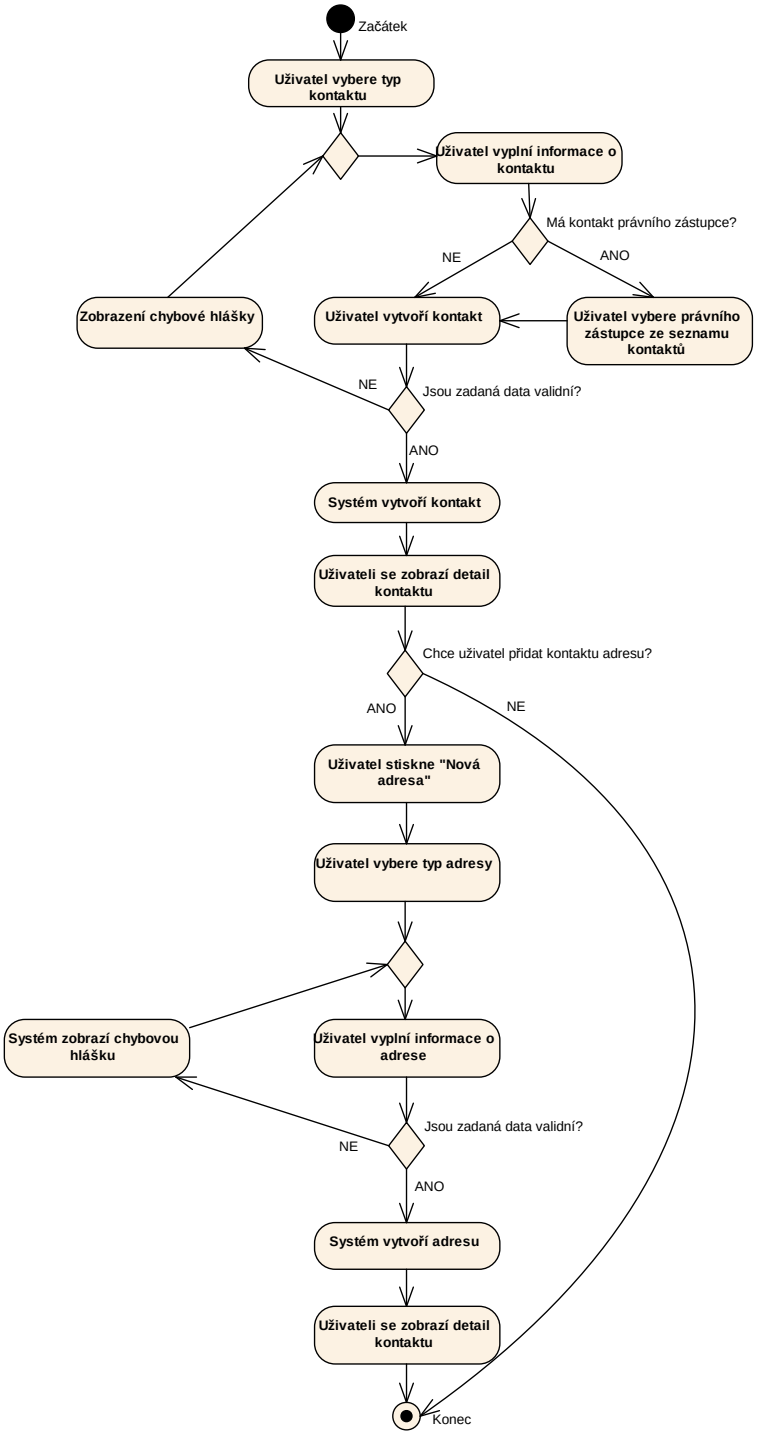
3. **Smazání typu advokátního úkonu:** Permanentně smaže typ advokátního úkonu z databáze.

4. **Úprava typu advokátního úkonu:** Umožňuje upravit název a koeficient typu advokátního úkonu.

Ostatní

1. **Zobrazení historie:**
   Tabulky: history
   Zodpovědnost: Advokát, Administrátor
   Historie změn v aplikaci. Při každém vytvoření, úpravě, nebo smazání záznamu v aplikaci je vložen záznam do tabulky history včetně uživatele, který tuto změnu provedl. Taktéž se do historie ukládá, když se kterýkoliv uživatel přihlásí, nebo odhlásí.

Poznámka 2.1 Obnovou smazaných spisů, kontaktů, dokumentů a uživatelů se systém ADVOKATIS nezabývá a pro jejich obnovení je potřeba upravit atribut `deleted` přímo v databázi.



Obrázek 3: Diagram aktivit pro vytvoření nového kontaktu a připojení adresy

2.6 Stavová analýza

V aplikaci ADVOKATIS lze u spisů, kontaktů, uživatelů a dokumentů uchovávat dva typy stavů: aktivní a smazaný. U schůzek je potřeba rozlišit, zda již proběhla, neproběhla nebo byla zrušena. Taktéž je potřeba rozlišovat stav jednotlivých úkolů.

- **Stav spisu, kontaktu, uživatele a dokumentu:** Tento stav je v databázových tabulkách `document`, `contact`, `user` a `file` uložen jako atribut `deleted`.
  1. **aktivní:** Atribut `deleted` je nastaven na 0. Spis, kontakt, uživatel, resp. dokument se zobrazuje uživatelům.
  2. **smazaný:** Atribut `deleted` je nastaven na 1. Spis, kontakt, uživatel, resp. dokument se nezobrazuje uživatelům. Smazaný záznam nelze nijak upravovat a jeho název je zbarven červeně.
- **Stav schůzky:** Tento stav je uložen v databázové tabulce `meeting_state`.
  1. **neproběhlá:** Značí, že tato schůzka ještě neproběhla. Pokud datum této schůzky již proběhlo je zbarveno červeně.
  2. **proběhlá:** Značí, že tato schůzka již proběhla a je zbarvena žlutě.
  3. **zrušená:** Značí, že tato schůzka byla zrušena a je také zbarvena žlutě.
- **Stav úkolu:** Tento stav je uložen v databázové tabulce `task_state`.
  1. **aktivní:** Značí, že je tento úkol aktivní a lze ukončit. Pokud termín tohoto úkolu již proběhl je zbarven červeně.
  2. **ukončený:** Značí, že je tento úkol ukončený a nelze již znova ukončit. Je podbarven žlutě.

2.7 Nefunkční požadavky

1. Aplikace bude primárně přístupná přes lokální síť. V případě potřeby je možné se do aplikace připojit i přes internet.
2. Aplikace bude implementována v jazyce PHP s použitím Zend Frameworku.
3. Aplikace bude využívat databázi typu MySQL.
4. Aplikace bude použitelná v moderních prohlížečích.
5. Aplikace bude využívat responzivního vzhledu, a tak bude funkční i na mobilních zařízeních.
6. Aplikace bude nasazena na počítači s OS Windows (XP a vyšší).

3 Použité technologie

V rámci této kapitoly jsou popsány technologie a nástroje, které byly použity při vývoji IS ADVOKATIS.

- **PHP 5.5.8**
  PHP [10] je multiplatformní skriptovací programovací jazyk, který obsahuje mnoho funkcí pro tvorbu webových aplikací s dobrou podporou databázových systémů a internetových protokolů, podrobnou dokumentací a velmi obsáhlou komunitou uživatelů.

- **MySQL**
  MySQL [11] je multiplatformní databázový systém, se kterým se komunikuje pomocí jazyka SQL. Pro informační systém ADVOKATIS byl vybrán kvůli jeho stabilitě. Podpora MySQL je taktéž v PHP velmi dobrá.

- **HTML5**
  Značkovací jazyk pro vytváření webových stránek. Podrobnější popis toho jazyka lze nalézt v knize *HTML5 cookbook* [4].

- **CSS**
  Jazyk, který určuje, jak se mají jednotlivé HTML, XHTML a XML elementy vykreslovat. Často je označován slovem kaskádové styly, protože na sebe mohou vrstvit definice stylu a vždy bude platit ta poslední. Podrobnější popis tohoto jazyka lze nalézt v knize *CSS cookbook* [3].

- **JavaScript**
  Jedná se objektově orientovaný skriptovací jazyk, který se stará o dynamičnost jednotlivých stránek. Nejčastěji jsem jej využil při odesílání a přijímání konkrétních dat pomocí AJAXu, kdy je možné např. upravit kontakt, aniž bych při zasílání požadavku načetl stránku znova. Popis této technologie lze nalézt v knihách *Pro Jquery* [5] a *PHP jQuery cookbook* [7].

- **Twitter Bootstrap**
  Twitter Bootstrap [12] je front-end framework, který využívá technologií HTML, JavaScript a kaskádových stylů (CSS). Umožňuje sjednotit grafickou stránku aplikace do jednoho stylu tak, aby byla správně zobrazitelná na všech zařízeních (responzivní design). Pro jeho využití v aplikaci stačí naimportovat kaskádový styl a Javascriptovou knihovnu do layoutu informačního systému. IS ADVOKATIS je celý vykreslován pomocí tohoto frameworku. Je založen na tzv. Grid systému [14], kdy se definují velikosti jednotlivých DIVů tak, aby zabíraly určitou část obrazovky podle aktuálního rozlišení a v případě potřeby se samy poskládaly pod sebe (např. u menších rozlišení). Např. v případě, že budu chtít šířku prvku při vysokém rozlišení přes polovinu obrazovky a zároveň při zmenšení rozlišení budu chtít DIV roztáhnout na celou šířku, tak mu nastavím atribut `class` na hodnotu `col-md-12`

`col-lg-6` (maximální hodnota je 12, která značí 100% šířku). Díky tomuto frameworku je možné implementovat grafickou stránku aplikace velmi rychle, aniž bych musel definovat své vlastní kaskádové styly. Dále framework obsahuje nejrůznější styly formulářů, vyskakovacích oken založených na JavaScriptu, či předem definovaných ikon.

- **Zend Framework**
  Zend Framework [13] je objektově orientovaný framework pro jazyk PHP. Byl zvolen kvůli jeho obsáhlosti, bezpečnosti a podpoře Model-View-Controller architektury. Framework obsahuje širokou škálu odladěných PHP funkcí, které byly využité při vývoji systému ADVOKATIS. V dnešní době existuje i Zend Framework 2, ale díky jeho menší komunitě jsem zvolil první verzi.

  Zend Framework se skládá z komponent, které je navíc možné kombinovat s jinými frameworky (např. s frameworkem Nette [16] nebo Symfony [17]).

  Přehled nejčastěji používaných komponent:

  - `Zend_Controller` (práce s controllery)
  - `Zend_Db` (práce s databází)
  - `Zend_Acl, Zend_Auth` (autorizace a autentizace uživatele)
  - `Zend_Form` (práce s formuláři)
  - `Zend_Locale` (práce s jazykovou verzí aplikace)
  - `Zend_Pdf` (pro generování PDF dokumentů)

  Kompletní seznam komponent dostupných pro Zend framework lze najít na adrese http://framework.zend.com/manual/1.5/en/requirements.zendcomponents.html

- **Ostatní front-end knihovny**
  IS ADVOKATIS využívá mnoho CSS a Javascriptových knihoven, které ve většině případů fungují tak, že se jejich CSS a Javascriptové soubory vloží do layoutu, a poté se na konkrétní HTML prvky pouze přidá patřičný atribut `class` nebo `id`. V následující části popíšu ty nejdůležitější. Všechny tyto knihovny obsahují licenci MIT a jsou tedy volně použitelné.

  - **SB Admin 2**
    SB Admin 2 [18] je grafická nástavba Twitter Bootstrapu.
  - **Bootstrap 3 Datepicker v4**
    Bootstrap 3 Datepicker v4 [19] je knihovna umožňující vložit do formulářového prvku ovládací prvek pro výběr data a času.
  - **DataTables**
    Knihovna DataTables [20] aplikuje na tabulky nejrůznější Javascriptové metody, které umožňují v tabulkách dynamicky vyhledávat, stránkovat, nebo vytvářet vlastní řadící algoritmy.

  - **Select2**
    Select2 [21] je knihovna umožňující vložit do formulářového prvku typu Select dynamické vyhledávání v jednotlivých položkách.
  - **Pnotify**
    Pnotify [22] je knihovna zobrazující malá vyskakovací okna, která jsou v aplikaci ADVOKATIS využívána pro informování uživatele, zda daná akce proběhla úspěšně, nebo neúspěšně.

- **MySQL Workbench 6.2**
  Pro práci s databázovým systémem MySQL byl zvolen program MySQL Workbench 6.2 [24], který lze zdarma stáhnout. Dle mého názoru je program MySQL Workbench velmi uživatelsky přívětivý a umožňuje veškerou správu systému MySQL.

- **EasyPHP 14.1**
  Aplikace byla vyvíjena na domácím počítači se systémem Windows 8.1, který je v řadách advokátů zdaleka nejpoužívanější. Vývoj byl možný díky samoinstalačnímu balíčku EasyPHP 14.1 [25], který obsahuje Apache 2.4.7, PHP ve všech verzích a MySQL 5.6.15. Taktéž obsahuje nástroj PhpMyAdmin [15] pro práci s databázovým systémem MySQL. Tento nástroj ovšem při vývoji aplikace ADVOKATIS nebyl využit. Díky této aplikaci lze provozovat vlastní PHP skripty na domácím počítači, aniž by bylo potřeba používat webový hosting. Instalace EasyPHP je na systému Windows velmi jednoduchá a případná konfigurace taktéž, díky jeho grafickému rozhraní.

- **Netbeans 8.0.1**
  Pro implementaci PHP aplikace bylo zvoleno vývojové prostředí Netbeans 8.0.1 [26], které obsahuje veliké množství potřebných pluginů, hlavně zásuvný modul pro podporu Zend Frameworku. Program NetBeans sám vytvoří celou kostru MVC frameworku a dále napovídá jednotlivé PHP metody.

4 Implementace

Tato kapitola popisuje jednotlivé vrstvy (dle rozdělení MVC) aplikace ADVOKATIS.

4.1 Datová vrstva

Datová vrstva aplikace představuje veškerou práci s databází. Pro mapování databázových tabulek byla využita komponenta `Zend_Db_Table`.

4.1.1 Připojení k databázi

Pro připojení k databázi využívá Zend Framework komponentu `Zend_Db_Adapter`, která podporuje všechny nejvíce používané databázové systémy. Aby adaptér mohl pracovat s MySQL databází, využívá PHP PDO_MYSQL ovladače. Pomocí adaptéru se lze připojit k databázi vícero způsoby (v konstruktoru třídy `Zend_Db_Adapter`, definicí v konfiguračním souboru atd.). V aplikaci ADVOKATIS bylo použito připojení skrze konfigurační soubor `configuration.ini`, který se nachází ve složce /application/ configs.

```
1 resources.db.adapter = PDO_MYSQL
2 resources.db.params.host = 127.0.0.1:3306
3 resources.db.params.username = root
4 resources.db.params.password = xxx
5 resources.db.params.dbname = advokatis
6 resources.db.params.charset = "utf8"
7 resources.db.isDefaultTableAdapter = true
```

Výpis 1: Zdrojový kód z konfiguračního souboru pro připojení k databázi

V předchozím výpisu kódu č. 1 značí řádek č. 1 výběr typu databáze, ke které se připojujeme. V řádcích č. 2-5 se zadávají připojovací údaje k databázi a poslední řádek udává, že se jedná o výchozí adaptér (v případě, že bychom se v aplikaci chtěli připojovat k různým databázím). Podrobnější popis připojení se k databázi pomocí Zend frameworku lze najít v knize *Pro Zend Framework techniques* [2].

4.1.2 Objektově relační mapování

Třída `Zend_Db_Table` páruje databázové tabulky na patřičné modely a je implementována pomocí návrhového vzoru Table Data Gateway[3]. Každou tabulku musí reprezentovat právě jeden model. Komponenta obsahujé také metody `find()` a `fetchAll()`, které byly využity při vývoji systému. Metoda `find($id)` vrací objekt typu `Zend_Db_Table_Rowset_Abstract`, kde parametr `$id` reprezentuje primární klíč záznamu v hledané databázové tabulce. Metoda `fetchAll($select)` vrací taktéž objekt typu `Zend`

[3]Návrhový vzor, kdy je každá databázová tabulka reprezentována samostatnou třídou. Tato třída poté provádí veškeré CRUD operace (INSERT, CREATE, UPDATE, DELETE).

_Db_Table_Rowset_Abstract, který na rozdíl od find() může obsahovat i více záznamů z databázové tabulky. Parametr $select obsahuje specifičtější databázový dotaz, např. WHERE, LIMIT atd. Všechny modely jsou v aplikaci uloženy ve složce /application/models/DbTable

```
1  class Application_Model_DbTable_Contact extends Zend_Db_Table_Abstract
2  {
3      protected $_name = 'contact';
4      protected $_primary = 'id';
5
6      public function fetchAllOrdered()
7      {
8          $select = $this->select()
9          ->from(array('c' => 'contact'))
10         ->where('c.deleted != ?', 1)
11         ->order("c.surname");
12         return $this->fetchAll($select);
13     }
14 }
```

Výpis 2: Zdrojový kód pro implementaci modelu

Ve zdrojovém kódu č. 2 je ukázka implementace modelu pro reprezentaci databázové tabulky contact, která reprezentuje kontakty. Metoda fetchAllOrdered() vrací veškeré nesmazané kontakty z tabulky contact. Výsledná data jsou seřazena vzestupně podle příjmení.

4.1.3 Transakce

V aplikaci ADVOKATIS jsou využívány také transakce. Např. v případě vytváření nového spisu, který se do databáze vloží až poté, co se uloží všechny jeho protistrany.

```
1  public function insertData($data, $opponents)
2  {
3      $dbAdapter = Zend_Db_Table::getDefaultAdapter();
4      $dbAdapter->beginTransaction();
5      try
6      {
7          document->insert($data);
8          if (document insert success) //pokud se podařilo vložení dat do tabulky document
9          {
10             for ($index <- 0 to délka[$opponents])
11             {
12                 document_has_opponent->insert($opponents[$index]);
13                 if (document_has_opponent insert fail)
14                 { //chyba při vkládání do tabulky document_has_opponent
15                     $dbAdapter->rollback();
16                     return "Chyba v tabulce Document_Oponent";
17                 }
18             }
```

```
19          } else {
20              $dbAdapter->rollback();
21              return "Chyba v tabulce Document";
22          }
23          $dbAdapter->commit();
24          return $ret;
25      }
26      catch (Exception $e)
27      {
28          $dbAdapter->rollback();
29          return $e->getMessage();
30      }
31  }
```

Výpis 3: Pseudokód metody insertData() pro ukázku práce s transakcemi

Ve zdrojovém kódu č. 3 lze vidět příklad práce s transakcemi. Pro využití transakcí v modelu je potřeba si načíst aktivní databázový adaptér, nad kterým se pak transakce budou spouštět. Ke spuštění transakce slouží metoda `beginTransaction()`. Dalšími metodami pro práci s transakcemi jsou: `commit()` pro uložení změn a `rollback()` pro zahození změn. Za použití transakce tedy metoda `insertData()` buď vytvoří záznam v tabulkách `document` i `document_has_opponent` nebo ani v jedné z nich. Podrobnější popis transakcí za využití Zend frameworku lze najít v knize *Zend Framework: programujeme webové aplikace v PHP* [1].

4.2 Aplikační vrstva

Tato vrstva je v aplikaci ADVOKATIS zastupována controllery. Každý controller se stará o veškerou logiku dané třídy. Controller se skládá z metod, které se v zendu nazývají akce. Příkladem může být akce zobrazení všech aktivních spisů. Controller, který obstarává práci se spisy, dostane od uživatele požadavek na výpis všech spisů. Controller tuto událost odchytí a vybraná akce požádá konkrétní model o patřičná data. Poté, co je model odešle zpět, controller data zpracuje a pošle do patřičného view (pohledu), který uživateli zobrazí výsledek. V následují části budou popsány jednotlivé controllery aplikace ADVOKATIS, příklad jednoduché akce vykonávané controllerem, vysvětlení, jak v Zend Frameworku fungují formuláře a na konci bude popis a řešení konkrétních problémů.

4.2.1 Controllery aplikace ADVOKATIS

- **DocumentController**
  Tento controller obsluhuje veškerou logiku spojenou se spisy.
- **ContactController**
  Tento controller obsluhuje veškerou logiku spojenou s kontakty.
- **AresController**
  Tento controller obsluhuje veškerou logiku spojenou s veřejnými rejstříky jako jsou Administrativní registr ekonomických subjektů a Portál veřejné správy.

- **CalendarController**
  Tento controller obsluhuje veškerou logiku spojenou s kalendářem.
- **UserController**
  Tento controller obsluhuje veškerou logiku spojenou s uživateli.
- **SettingsController**
  Tento controller obsluhuje veškerou logiku spojenou s nastavením systému, správy číselníků a historií.
- **MeetingController**
  Tento controller obsluhuje veškerou logiku spojenou se schůzkami.
- **TaskController**
  Tento controller obsluhuje veškerou logiku spojenou s úkoly.
- **TermController**
  Tento controller obsluhuje veškerou logiku spojenou s termíny k dokumentům.
- **AuthController**
  Tento controller obsluhuje veškerou logiku spojenou s přihlašováním jednotlivých uživatelů do systému.
- **IndexController**
  Tento controller obsluhuje logiku spojenou se zobrazením úvodní stránky aplikace. Dále obsluhuje hlavní vyhledávací formulář a poznámkový blok každého uživatele.
- **ErrorController**
  Tento controller obsluhuje logiku spojenou s chybovými hlášeními.

Jednotlivé akce byly pojmenovány tak, aby bylo patrné, co mají na starost. Akce, které mají na konci jména slovo ajax, obsluhují XMLHttpRequest požadavky posílané AJAXem, tedy obslouží konkrétní požadavek, aniž by byla stránka znovu načtena (asynchronní výměna dat). Podrobnější popis XMLHttp požadavků lze najít v knize *Professional Ajax* [6].

4.2.2 Jednoduchá akce

Každý controller se musí skládat z akcí, což jsou metody zakončené slovem Action. Pro každou akci musí existovat view se stejným názvem, jako je název akce tak, aby nebyla potřeba při každém odesílání dat z controlleru do view definovat o jaký view se jedná (např. v controlleru `Contact` se zavolá akce `detail` která, když vykoná požadavek odešle data do svého view, který se nachází ve složce /application/views/scripts/contact/detail.phtml).

```
1  class TaskController extends Zend_Controller_Action
2  {
3      public function allAction ()
4      {
5          $tasks = new Application_Model_DbTable_Task();
6          $loggedUser = Zend_Auth::getInstance()->getStorage()->read();
7          $this->view->tasks = $tasks->fetchAllComplete();
8          $this->view->userid = $loggedUser->id;
9          $form_edit_task = new Application_Form_EditTask();
10         $this->view->form_edit_task = $form_edit_task;
11     }
12 }
```

Výpis 4: Zdrojový kód akce all v Task controlleru

Předchozí kód č. 4 ukazuje controller `Task`, který obsluhuje veškerou logiku spojenou s úkoly a který má jen jednu akci (tento controller obsahuje více akcí, ale protože se jedná o ukázkový příklad, byla vypsána pouze jedna), která odesílá seznam všech úkolů do patřičného view. Pro definování nového controlleru je potřeba vytvořit třídu dědící od třídy `Zend_Controller_Action`. Akce `allAction()` inicializuje nový model, uloží si data o momentálně přihlášeném uživateli a poté, zavoláním `$this->view->tasks=$tasks ->fetchAllComplete()` odešle data z modelu do svého view. V další části kódu odešle do view také data o přihlášeném uživateli a inicializuje nový formulář pro úpravu úkolů, který následně odešle také do view.

4.2.3 Formuláře

Formuláře v Zend frameworku reprezentuje třída `Zend_Form`, která obsahuje mnoho typů validátorů[4] (přehled validátorů je na adrese http://framework.zend.com/manual/1.12/en/zend.validate.set.html). Všechny formuláře se nacházejí ve složce /application/forms.

```
1  class Application_Form_AddDocument extends Zend_Form
2  {
3      public function init ()
4      {
5          $this->setMethod('post');
6          $name = new Zend_Form_Element_Text('name');
7          $name->addValidator('stringLength', false, array(3, 100));
8          $name->setRequired(true);
9          $name->class = 'form-control';
10         $name->setLabel('* Název:');
11         $name->setAttrib('placeholder', 'název/popis spisu');
12
13         $submit = new Zend_Form_Element_Submit('add_doc_submit');
```

[4]Třídy a metody, které kontrolují správnost dat ve formuláři podle nadefinovaných pravidel. Např. použitím validátoru Zend_Validate_StringLength mohu nadefinovat, že konkrétní prvek formuláře musí mít patřičnou délku, jinak mi tento validátor při validaci formuláře vrátí chybu.

```
14          $submit->class = 'btn btn-success';
15          $submit->setLabel('Vytvořit');
16
17          $this->addElements(array($name,$submit));
18
19          $this->addElement('hash', 'no_csrf_foo', array('salt' => 'unique'));
20      }
21  }
```

Výpis 5: Zdrojový kód jednoduchého formuláře pro vytvoření nového spisu

Zdrojový kód č. 5 ukazuje definici jednoduchého formuláře `AddDocument`, který představuje formulář pro vytvoření nového spisu. Na začátku je potřeba definovat, jakou metodou má data odesílat (POST/GET). Proměnná `$name` definuje formulářový prvek typu Text, na který je aplikován validátor pro ověření délky jeho obsahu. Metoda `setRequi red(true)` říká, že je tento prvek povinný. Aby formulář úspěšně prošel validací (ukázka kódu pro validaci formuláře v controlleru se nachází dále), je potřeba, aby uživatel vyplnil toto textové pole (nenechal je prázdné). Dále obsahuje definici HTML atributu class, který mu bude vykreslen ve view. Proměnná `$submit` definuje prvek typu Submit, který představuje tlačítko pro odesílání formuláře. Metoda `addElements()` říká formuláři, které prvky formuláře jsou aktivní (je to z toho důvodu, že lze použít jeden formulář, jak pro vytváření spisu, tak i pro upravení spisu, kdy se formuláři např. nastaví, aby deaktivoval určité prvky). Poslední řádek kódu přidává neviditelný prvek (token), který zabraňuje CSRF útokům (typy útoků jsou popsány v kapitole 5) a zabraňuje, aby byl formulář odeslán vícekrát.

Zend framework umožňuje v případě potřeby nadefinovat své vlastní validátory. Každopádně obsahuje velké množství již hotových validátorů, a proto v aplikaci ADVOKATIS nebyla potřeba implementovat žádný vlastní.

4.2.4 Poznámkový blok

Většinu uživatelských požadavků aplikace ADVOKATIS obsluhuje dynamicky, aniž by se pro vykonání akce musela aktuální stránka znovu načíst. Tyto požadavky jsou do controlleru odeslány pomocí AJAXu a akce na ně odpovídá JavaScriptovým datovým formátem JSON. Zda jsou data odeslána do controlleru AJAXem lze zjistit metodou `isXmlHttp Request()`, která vrací `TRUE/FALSE`. Příklad takové AJAX akce ukazuje následující zdrojový kód č. 6 metody pro ukládání uživatelova poznámkového bloku.

```
1   public function usernoteajaxAction()
2   {
3       $user_note_form = new Application_Form_UserNote();
4       $loggedUser = Zend_Auth::getInstance()->getStorage()->read();
5       $users = new Application_Model_DbTable_User();
6
7       if ($this->getRequest()->isXmlHttpRequest())
```

```
 8      {
 9          if ($this->getRequest()->isPost())
10          {
11              $data = $this->getRequest()->getPost();
12              if ($user_note_form->isValid($data))
13              {
14                  $loggedUser->note = $data['user_note'];
15                  $res = $users->updateData(array('note'=>$data['user_note']),$loggedUser->id);
16                  //kontrola, zda byla data upravena a vrácení
17                  //příslušné odpovědi metodou Zend_Json::encode
18              }
19              else
20              {
21                  //vrácení chybových hlášení z validátorů metodou Zend_Json::encode
22              }
23          }
24          else
25          {
26              //pokud neobdržel POST, vrátí chybové hlášení metodou Zend_Json::encode
27          }
28      }
29      else
30      {
31          //pokud se nejedná o AJAX, vrátí chybové hlášení metodou Zend_Json::encode
32      }
33  }
```

Výpis 6: Zdrojový kód AJAX akce pro obsluhu poznámkového bloku

Každý přihlášený uživatel vidí na úvodní straně aplikace textové pole, do kterého si může napsat rychlou poznámku (formulář `UserNote`). Pomocí Javascriptové metody `Blur()` lze rozpoznat, že uživatel opustil toto textové pole (dopsal poznámku), a poté automaticky AJAXem poslat obsah poznámkového bloku do akce `usernoteajaxAction()`. Tato akce vytvoří novou instanci formuláře, zjistí si informace o aktuálně přihlášeném uživateli, kterému poznámkový blok patří a vytvoří instanci modelu `User`. Dále, pokud tato akce přijme požadavek typu XmlHttpRequest, pokusí se získat novou uživatelovu poznámku, zkontroluje, zda proběhla nějaká změna oproti poznámce minulé, případně se pokusí uložit tuto poznámku do databáze. V případě jakékoliv chyby odesílá zpět chybovou odpověď v datovém formátu typu JSON. Více o Javascriptové metodě `blur()` lze nalézt v knize *Pro Jquery* [5]. Detailnější popis AJAX akcí je ukázán v knize *PHP jQuery cookbook* [7].

4.2.5 Jeden vyhledávací formulář pro spisy i kontakty

Veškeré vyhledávání v aplikaci ADVOKATIS je kdykoliv dostupné, protože vyhledávací formulář je umístěn v levé horní části, hned nad navigací. O vyhledávaní se stará controller `Index` a jeho akce `search`. Obsahuje jedno textové pole pro zadání hodnoty a dvě odesílací tlačítka (jedno na vyhledávání ve spisech a druhé pro kontakty).V případě, že uživatel stisknul tlačítko pro vyhledávání ve spisech, je odeslaná hodnota porovnávána

se sadou regulárních výrazů a vyhodnoceno, zda je vstup ve správném formátu. Pokud ano, je zavolána patřičná metoda modelu `Document` pro získání dat. Pokud uživatel vyhledává v kontaktech, tak se odeslaná hodnota taktéž porovnává se sadou regulárních výrazů, ale metody se volají nad modelem `Contact`. V následující části budou popsány jednotlivé formáty vstupu, které aplikace umí vyhodnotit. V obou případech jsou vstupní data ošetřena o mezery na konci a na začátku PHP metodou `trim()`. O porovnávání dle regulárních výrazů se stará PHP metoda `preg_match()`, která vrací `TRUE/FALSE`. Jako první parametr se do ní zadává regulární výraz a jako druhý parametr proměnná, ve které se má porovnávat.

- **Vyhledávání ve spisech**
  1. **podle roku (např. "/15")**
     Vyhledá všechny spisy, které patří do tohoto roku.
     Regulární výraz: `"/^\/[0-9]+$/"`
  2. **podle čísla a roku (např. "1/14")**
     Vyhledá konkrétní spis.
     Regulární výraz: `"/^[0-9]+\/[0-9]+,?$/"`
  3. **podle konkrétních spisů oddělených čárkou (např. "1/14, 8/15, 4/14")**
     Vyhledá konkrétní spisy.
     Regulární výraz: `"/^([0-9]+\/[0-9]+,)+([0-9]+\/[0-9]+,?)$/"`
- **Vyhledávání v kontaktech**
  1. **podle rodného čísla (např. "800519/1668")**
     Vyhledá kontakt se zadaným rodným číslem. Lze zadat i ve formátu bez lomítka.
     Regulární výraz: `"/^[0-9]{6}\/?[0-9]{3}[0-9]?$/"`
  2. **podle data narození (např. "19.5.1980")**
     Vyhledá všechny kontakty narozené v zadaný den.
     Regulární výraz: `"/^[0-9]{1,2}\.[0-9]{1,2}\.[0-9]{4}$/"`
  3. **podle IČ (např. "16645689")**
     Vyhledá kontakt se zadaným IČ.
     Regulární výraz: `"/^([0-9]){8}$/"`
  4. **podle jména a příjmení (např. "Jan Novák")**
     Vyhledá kontakty se zadaným jménem a příjmením. Lze zadat v pořadí jméno, příjmení, nebo v opačném pořadí příjmení, jméno.
     Regulární výraz: `"/^[\p{L}-]{2,25} [\p{L}-]{2,25}$/u"`
  5. **podle jména, nebo příjmení (např. "Novasoft")**
     Vyhledá kontakt se zadaným jménem, nebo příjmením.
     Regulární výraz: `"/^[\p{L}-]{2,50}$/u"`

V případě, že vyhledávaný výraz neodpovídá ani jedné z předchozích podmínek, je uživateli zobrazeno chybové hlášení.

4.2.6 Komunikace s veřejnými rejstříky

Aplikace ADVOKATIS umí při vytváření nového kontaktu automaticky doplnit jeho detailní informace (Administrativní registr ekonomických subjektů) a datovou schránku (Portál veřejné správy) dle zadaného IČ. O tuto komunikaci se stará AJAX akce ares v controlleru Ares a využívá PHP knihovny cURL [23], která dokáže přenášet data dle zadané URL adresy. V takto stažené stránce lze vyhledávat jako v XML souboru, např. jazykem XPath.

- **Administrativní registr ekonomických subjektů** [30]
  IČ je vloženo do URL adresy a poskytuje kompletní XML odpověď.
  Vzor URL: `"http://wwwinfo.mfcr.cz/cgi-bin/ares/darv_bas.cgi?ico=".$ic."&xml=0"`

- **Portál veřejné správy** [31]
  IČ je vloženo do URL adresy a poskytuje odpověď ve formě HTML stránky.
  Vzor URL: `"https://seznam.gov.cz/ovm/searchList.do?ref=obcan&start=null&searchCriterium=ovm_ico_of_subject&searchValue=".$ic`

4.3 Prezentační vrstva

Tato vrstva je v aplikaci ADVOKATIS zastupována pomocí view (pohled). Každá akce v controlleru by měla mít svůj pohled ve složce /application/views/scripts (neplatí pro AJAX akce). Pohledy v MVC architektuře vykreslují výsledné HTML stránky a naplňují je daty získanými z controlleru. Tuto vrstvu v Zend frameworku reprezentuje třída `Zend_View`.

4.3.1 Vykreslování formulářů

Každý formulář, který je akcí odesílán do pohledu lze jednoduše vykreslit příkazem `$this->form`, kde `$this` představuje instanci `Zend_View` a `form` název, pod kterým je formulář poslán do pohledu. Každopádně, aby formulářové prvky vypadaly správně podle stylu Twitter Bootstrapu, je potřeba jednotlivým prvkům formuláře přidat správné atributy. Proto byl pro vlastní účely vytvořen následující plugin (zdrojový kód č. 7) pro vykreslování jednotlivých formulářových prvků tak, aby odpovídaly stylu Twitter Bootstrapu.

```
1  function printFormElement($element, $datepicker = NULL)
2  {
3      print "<div class=' clearfix '>";
4      print "<div class=' fl-l clearfix '>";
5      print $element->renderLabel();
6      print "</div>";
7      print "<div class='element-error-message font-size-8 fl-r clearfix color-red'>";
8      if (count($element->getMessages()) > 0)
9      {
```

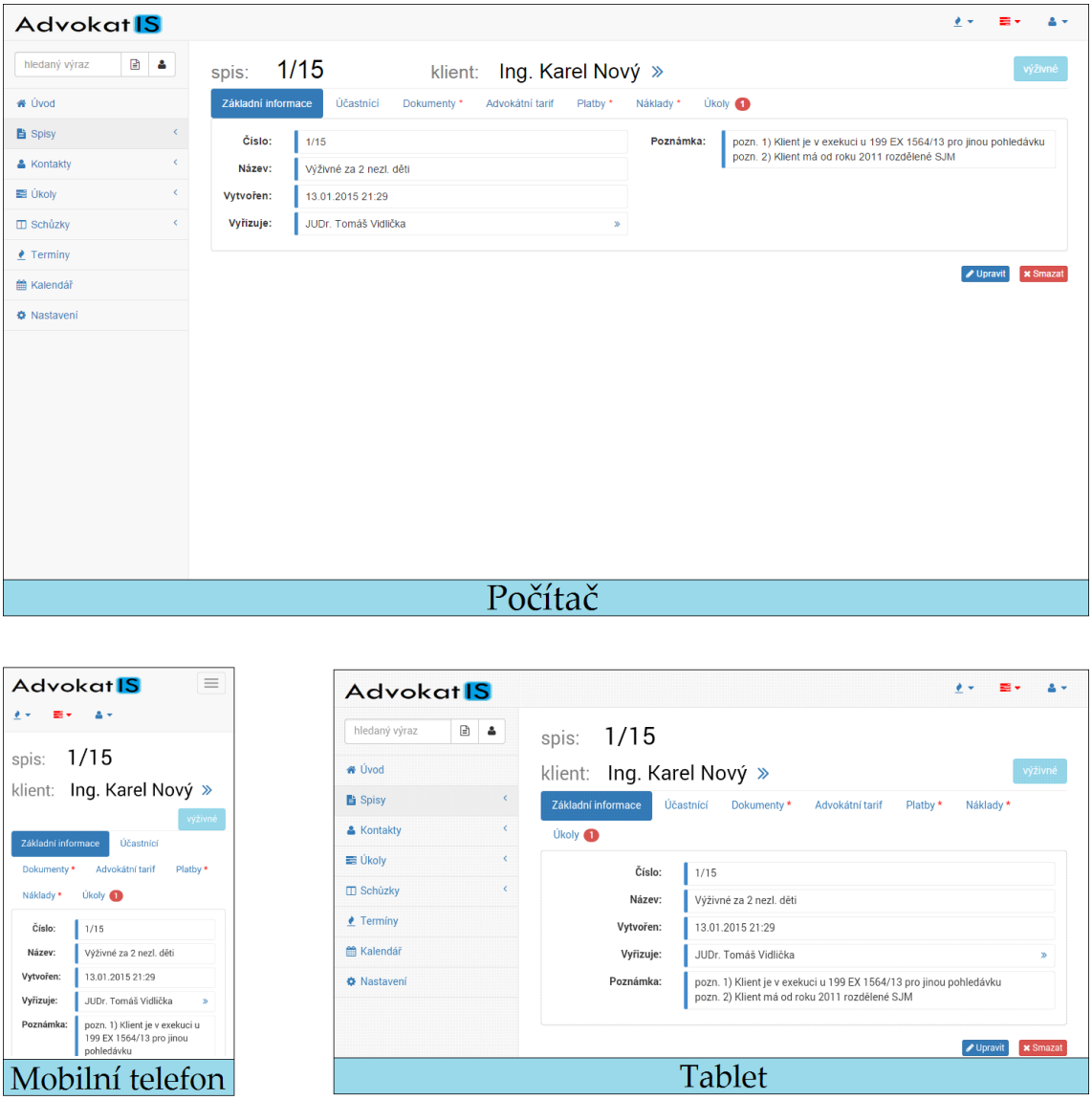
```
10          print "<ul>";
11          foreach ($element->getMessages() as $key => $value)
12          {
13              print "<li>" . $value . "</li>";
14          }
15          print "</ul>";
16      }
17      print "</div>";
18      print "</div>";
19      if ($datepicker != NULL)
20      {
21          print "<div class='input-group date' id='".$datepicker."'>";
22          print $element->renderViewHelper();
23          print "<span class = 'input-group-addon'><span class = 'glyphicon glyphicon-calendar'></
                span>";
24          print "</span>";
25          print "</div>";
26      }
27      else
28      {
29          print $element->renderViewHelper();
30      }
31  }
```

Výpis 7: Zdrojový kód metody printFormElement

Metoda `printFormElement()` přijímá v prvním parametru konkrétní formulářový prvek a jako druhý parametr (nepovinný) název atributu `class` pro určení, že má na tento prvek být ještě navíc aplikována knihovna Datepicker pro přidání ovládacího prvku pro výběr data a času.

4.3.2 Výsledný vzhled aplikace

Obrázek č. 4 ukazuje výsledný vzhled aplikace ADVOKATIS, zobrazený na různých zařízeních. Vždy se jedná o stejnou stránku detailu spisu. Např. na mobilním telefonu je navigace zcela schována a odkaz na její zobrazení se nachází v pravém horním rohu displeje. Popis jednotlivých položek z navigace lze nalézt v Uživatelské příručce, která se nachází v příloze D.



Obrázek 4: Zobrazení stejné stránky na různých zařízeních

5 Webový server

Jako webový server byla použita aplikace EasyPHP 14.1 [5]. Ve výchozím nastavení běží Apache server na portu 80. Poté, co je aplikace spuštěna, je možné dostat se do kořenové složky serveru Apache (většinou /EasyPHP/data/localweb/projects) zadáním adresy http://127.0.0.1. V případě, že aplikace běží na samostatném serveru a uživatel se chce připojit, stačí zadat IP adresu serveru v místní síti.

V konfiguračním souboru `httpd.conf` (konfigurační soubor Apache serveru) bylo pro fungování Zend Frameworku potřeba povolit mód rewrite odkomentováním řádku `Load Module rewrite_module modules/mod_rewrite.so`, který umožňuje nejrůznější úpravy URL adres (hlavně umožňuje přesměrovávání a pozměnění adres).

Pro úplnou funkčnost aplikace bylo nutné v konfiguračním souboru `php.ini` (konfigurační soubor PHP) povolit následující vlastnosti:

- `file_uploads = On` (povolení nahrávání souborů na server)
- `post_max_size = 12M` (maximální velikost požadavků typu POST - hlavně kvůli velikosti nahrávaných příloh k jednotlivým dokumentům)
- `upload_max_filesize = 12M` (maximální velikost nahrávaných dokumentů na server, měl by se shodovat s `post_max_size`)
- `extension = php_fileinfo.dll` (povolení zjišťování informací o konkrétním nahrávaném dokumentu)
- `extension = php_curl.dll` (povolení knihovny cURL pro komunikaci s veřejnými rejstříky)

K aplikaci ADVOKATIS se lze také v případě potřeby připojit vzdáleně přes internet. Pro takovéto připojení bylo potřeba ve směrovači povolit přesměrovávání portů, a to konkrétně port 80 na adresu serveru v místní síti. Dále je potřeba v konfiguračním souboru `httpd.conf` povolit naslouchání na portu 80 odkomentováním příkazu `Listen *:80`. Ve výchozím nastavení Apache server po zadání IP adresy zobrazí adresář s jednotlivými aplikacemi, které jsou nahrány v jeho kořenovém adresáři. Aby byla práce pohodlnější, je dobré vytvořit si Virtual Host[6], který se postará o přesměrování na úvodní stranu aplikace ihned, co zaznamená požadavek na zobrazení kořenového adresáře Apache serveru.

[5] Návod pro zprovoznění EasyPHP lze nalézt na adrese http://www.easyphp.org/support/quickstart

[6] Způsob jak obsluhovat více domén, které běží na jednom serveru. Každá doména může tedy načítat data z jiného adresáře. Virtuální hostování umožňuje používat mod_rewrite, který se běžně využívá pro podstrčení nebo přesměrování webových stránek.

Aplikace ADVOKATIS je nyní spuštěna pouze pro testovací a prezentační účely na adrese http://advokatis.cz/public/ a lze se do ní přihlásit pomocí účtu **admin** a hesla **1234**.

5.1 Parametry webového serveru

- **OS:** Windows XP a novější
- **HDD:** 15 GB
- **RAM:** 512 MB

Jedná se o minimální požadavky na webový server. Pro úplnou plynulost aplikace doporučuji rychlejší počítač. Vývoj aplikace ADVOKATIS probíhal na počítači s čtyřjádrovým procesorem Intel i5, 8 GB RAM a OS Windows. Na tomto počítači byla aplikace naprosto svižná, jak při přístupu z místní sítě, tak z internetu. Aplikaci lze také provozovat na OS Linux, ale v takovém případě nelze použít program EasyPHP a je potřeba nainstalovat ručně Apache server, PHP a MySQL databázi.

5.2 Bezpečnost aplikace ADVOKATIS

Tato kapitola se zaměřuje na dva nejběžnější útoky na webové aplikace a jsou v ní popsány Zend komponenty, které svou funkcí přispívají k zabezpečení aplikace.

- **SQL injection**
  Tato technika napadá databázovou vrstvu aplikace tím, že do vstupu (např. formulářového prvku) vloží místo konkrétní hodnoty SQL kód, který se v databázi poté vykoná. Tento útok lze znemožnit nahrazením potencionálně nebezpečných znaků ve vstupních datech (tzv. escapování). Komponenty `Zend_Db` a `Zend_Select`, které se v aplikaci využívají pro práci s databází, automaticky tyto nebezpečné vstupy ošetřují. Pro zvýšení bezpečnosti bylo v aplikaci využito formulářových validátorů, které nebezpečné vstupy filtrují, ještě než jsou data poslána do databáze.
- **CSRF**
  Jde o typ útoku, kdy útočník odešle požadavek na server bez vědomí uživatele (např. podstrčení dat metodou GET konkrétní akci). Také se s tímto problémem lze setkat při odeslání formuláře a následném obnovení stránky, kdy se data odešlou podruhé. Tento útok lze v Zend Frameworku ošetřit využitím autorizačního tokenu s každým odeslaným formulářem. Tento token si vytvoří hash pro konkrétní uživatelovu akci a při kontrole validity formuláře jej porovná s hashem původním. V případě rozdílných hashů není formulář validní a vrátí chybovou hlášku.

Dále se o část bezpečnosti aplikace ADVOKATIS stará komponenta `Zend_Auth`, která při úspěšném přihlášení uživatele automaticky porovnává jeho přihlašovací údaje s databází a poté uloží jeho identitu do session. Uživatelova identita je kontrolována při inicializaci každého controlleru v aplikaci, čili při volání jakékoliv akce a v případě, že uživatel nemá identitu, je okamžitě přesměrován na přihlašovací stránku. Zdrojový kód této kontroly a případného přesměrovaní lze vidět ve výpise č. 8.

```
1  public function init ()
2  {
3      if (!Zend_Auth::getInstance()->hasIdentity())
4      {
5          $this->_helper->redirector("index", "auth");
6      }
7  }
```

Výpis 8: Zdrojový kód pro kontrolu identity a případné odhlášení z aplikace

Dalším bezpečnostním prvkem aplikace je kontrola přístupových práv jednotlivých uživatelů. O rozlišování uživatelských rolí a jejich práv se stará komponenta `Zend_Acl`, která spolupracuje s komponentou `Zend_Auth`, ze které získá roli přihlášeného uživatele. Konkrétní práva pro jednotlivé uživatele jsou nastavena v souboru `Bootstrap.php`. Každé roli se zakáže přístup ke všem akcím (příkaz `$acl->deny()`), a poté se každé roli povolí jednotlivé akce. Takto lze povolit či zakázat jak práci s celým controllerem, tak s jeho jednotlivými akcemi. Ukázkový kód jednoduchého nastavení přístupových práv se nachází ve výpise č. 9.

```
1  $acl = new My_Acl();
2  $acl->addRole('advokat');
3  $acl->addRole('sekretarka');
4  $acl->addResource('contact');
5  $acl->addResource('index');
6  $acl->addResource('document');
7  $acl->deny();
8  $acl->allow('advokat', 'index');
9  $acl->allow('advokat', 'contact');
```

Výpis 9: Zdrojový kód pro nastavení přístupových práv jednotlivým rolím

6 Testování

Aplikaci ADVOKATIS byla otestována dvěma způsoby, a to naplněním databáze velikým množstvím automaticky generovaných dat a zkušebním provozem v Advokátní kanceláři JUDr. Jany Mikulové.

6.1 Testování datové vrstvy

Pro tento test byl využit program EMS Data Generator for MySQL [27], který po připojení k MySQL databázi naplní vybrané tabulky automaticky generovanými daty. Tímto způsobem byly nejčastěji využívané tabulky (spisy, kontakty, adresy, schůzky, úkoly, historie) naplněny řádově tisíci zkušebních dat a pomocí webového pluginu Firebug 2.0.9 [28] se měřil čas zpracování jednotlivých stránek. Aplikace byla svižná, jednotlivé stránky se načítaly v průměru 250ms. Pouze u stránek, kde je zobrazen kompletní seznam položek (např. všechny spisy, všechny kontakty) trvalo načtení stránky zhruba 2s. Je to z toho důvodu, že na těchto stránkách byla použita Javascriptová knihovna Datatables, která jednotlivé položky po načtení řadí, stránkuje a umožňuje v nich dynamicky vyhledávat. Takový případ v reálném nasazení nenastane, neboť menší advokátní kanceláře mají roční nápad 100-200 nových spisů a tento test probíhal s 10000 spisy.

Test probíhal na čtyř-jádrovém počítači, jehož parametry jsou detailněji popsány v kapitole Webový server 5.1.

6.2 Testování v reálném prostředí

Poté, co byla aplikace ADVOKATIS naimplementována, byla zkušebně nasazena v Advokátní kanceláři JUDr. Jany Mikulové. Databáze aplikace byla naplněna pouze povinnými daty (administrátor pro prvotní vytvoření uživatelů, seznam zemí, číselníky - stavy spisů, schůzek, úkolů, typy odměn, advokátních úkonů, kontaktů, směru dokumentů a rolemi) jako při reálném nasazení. Tento test odhalil některé nedostatky a vedl k následujícím úpravám IS:

- Doplněna možnost upravení typu advokátního úkonu.
- Doplněna možnost evidovat datum úmrtí/zániku kontaktu.
- Upraveno, že termíny úkolů mohou být zadány pouze v čase, který ještě nenastal (nelze vytvořit úkol s termínem, který již proběhl).

7 Závěr

Ve své práci jsem navrhnul aplikaci pro menší advokátní kancelář, která se může skládat z advokáta, advokátního koncipienta, sekretářky a případně administrátora systému. Umožňuje evidovat veškeré spisy, kontakty, dokumenty, schůzky a úkoly, které jsou pro chod advokátní kanceláře nezbytné.

Aplikace ADVOKATIS je navržena tak, aby k ní mohlo být uživatelem přistupováno jak vzdáleně přes internet, tak lokálně přes místní síť. Této funkcionality jsem dosáhl použitím a vhodným nakonfigurováním Apache serveru. Tuto funkci aplikace považuji za největší výhodu v porovnání s konkurenčními systémy.

Aplikace využívá řadu moderních knihoven. Pravděpodobně nejdůležitější z nich je Twitter Bootstrap, který umožňuje,že aplikace bude zcela funkční na mobilních telefonech a tabletech. Primárně je však aplikace určena pro použití ve webovém prohlížeči počítače.

Podle nedostatků nalezených v již existujících konkurenčních systémech jsem aplikaci ADVOKATIS naimplementoval tak, aby maximálně vyhovovala advokátním kancelářím. Nejčastějším problémem v těchto systémech byla jejich nepřehlednost a nedostatečná funkčnost při prohlížení na mobilních zařízeních. Tyto chyby jsem v aplikaci ADVOKATIS vyřešil použitím vhodných knihoven.

Při vytváření této aplikace jsem se důkladně seznámil s PHP frameworkem Zend, který mohu využít i v dalších projektech. Odzkoušel jsem si kompletní analýzu, návrh a implementaci webového informačního systému. Také jsem si prohloubil znalosti ohledně HTML, CSS a Javascriptových knihoven, které jsem v této bakalářské práci hojně využíval.

V budoucnu mám v plánu aplikaci ADVOKATIS dále rozšiřovat a upravovat tak, aby vyhovovala jednotlivým advokátním kancelářím, které ji budou chtít používat. Již mám naplánováno několik funkcí, které by bylo možné do systému přidat. Jedná se např. o následující:

- Komunikace s dalšími veřejnými rejstříky
- Automatické generování dokumentů ve formátu pdf (např. plná moc, která bývá většinou stejná a liší se pouze informacemi o klientovi nebo vygenerování faktury)
- Možnost evidovat knihu jízd

Závěrem bych rád poděkoval Advokátní kanceláři JUDr. Jany Mikulové, se kterou jsem průběžně aplikaci konzultoval a jejíž advokátní kancelář moji aplikaci také otestovala.

8 Reference

[1] BÖHMER, Marian, *Zend Framework: programujeme webové aplikace v PHP*, Vyd. 1. Brno: Computer Press, 2010, 416 s. ISBN 978-80-251-2965-4.

[2] LYMAN, Forrest, *Pro Zend Framework techniques: build a full CMS project*, New York: Distributed to the book trade worldwide by Springer-Verlag, 2009, 240 s. ISBN 978-1-4302-1879-1.

[3] SCHMITT, Christopher, *CSS cookbook*, 3rd ed. Sebastopol, CA: O'Reilly, 2010, 702 s. ISBN 978-0-596-15593-3.

[4] SIMPSON, Christopher Schmitt and Kyle, *HTML5 cookbook*, 1st ed. Beijing: O'Reilly, 2011, 284 s. ISBN 978-1-449-39679-4.

[5] FREEMAN, Adam, *Pro Jquery*, Distributed to the book trade worldwide by Springer Science+Business Media, 2012, 985 s. ISBN 9781430240952.

[6] ZAKAS, Nicholas C, Jeremy MCPEAK a Joe FAWCETT, *Professional Ajax*, 2nd ed. Indianapolis, IN: Wiley, 2007, 598 s. ISBN 978-0-470-10949-6

[7] JOSHI, Vijay, *PHP jQuery cookbook: over 60 simple but highly effective recipes to create interactive web applications using PHP with jQuery*, Birmingham, U.K.: Packt Open Source, 2010, 317 s. ISBN 978-1-849512-74-9.

[8] *IS Jurisoft* [online]. URL: <http://jurisoft.cz/> [cit. 2015-04-16].

[9] *IS AdvocatusDigital* [online]. URL: <http://advocatusdigital.com/cz> [cit. 2015-04-16].

[10] *PHP* [online]. URL: <http://php.net/> [cit. 2015-04-18].

[11] *MySQL* [online]. URL: <https://www.mysql.com/> [cit. 2015-04-18].

[12] *Twitter Bootstrap* [online]. URL: <http://getbootstrap.com> [cit. 2015-04-18].

[13] *Zend Framework* [online]. URL: <http://framework.zend.com> [cit. 2015-04-20].

[14] *w3schools.com - Bootstrap Grid System* [online]. URL: <http://www.w3schools.com/bootstrap/bootstrap_grid_system.asp> [cit. 2015-04-29].

[15] *phpMyAdmin* [online]. URL: <http://www.phpmyadmin.net> [cit. 2015-04-20].

[16] *Nette Framework* [online]. URL: <http://nette.org/> [cit. 2015-04-20].

[17] *Symfony Framework* [online]. URL: <http://symfony.com/> [cit. 2015-04-20].

[18] *SB Admin 2* [online]. URL: <http://startbootstrap.com/template-overviews/sb-admin-2> [cit. 2015-04-20].

[19] *Bootstrap 3 Datepicker v4* [online]. URL: <http://eonasdan.github.io/bootstrap-datetimepicker> [cit. 2015-04-20].

[20] *DataTables* [online]. URL: <https://www.datatables.net> [cit. 2015-04-20].

[21] *Select2* [online]. URL: <https://fk.github.io/select2-bootstrap-css> [cit. 2015-04-20].

[22] *Pnotify* [online]. URL: <http://sciactive.github.io/pnotify> [cit. 2015-04-20].

[23] *Client URL Library* [online]. URL: <http://php.net/manual/en/book.curl.php> [cit. 2015-04-20].

[24] *MySQL Workbench* [online]. URL: <https://www.mysql.com/products/workbench> [cit. 2015-04-23].

[25] *EasyPHP* [online]. URL: <http://www.easyphp.org/> [cit. 2015-04-23].

[26] *Netbeans* [online]. URL: <https://netbeans.org/> [cit. 2015-04-18].

[27] *EMS Data Generator for MySQL* [online]. URL: <http://www.sqlmanager.net/en/products/mysql/datagenerator> [cit. 2015-04-24].

[28] *Firebug* [online]. URL: <http://getfirebug.com/> [cit. 2015-04-24].

[29] *Apigen* [online]. URL: <http://www.apigen.org/> [cit. 2015-04-19].

[30] *Administrativní registr ekonomických subjektů (Ares)* [databáze online]. Praha: Ministerstvo financí ČR, 1999. URL: <http://wwwinfo.mfcr.cz/> [cit. 2015-04-23].

[31] *Portál veřejné správy* [databáze online]. Praha: Ministerstvo vnitra ČR. URL: <https://portal.gov.cz/portal/obcan/> [cit. 2015-04-23].

A Databázové tabulky aplikace ADVOKATIS

- **document**
  Tabulka uchovávající informace o jednotlivých spisech, např. číslo a rok spisu, který kontakt zde vystupuje jako klient, název spisu, poznámka, který právník má spis na starost atd. V aplikaci je tato tabulka reprezentována modelem `Document.php`.

- **contact**
  Tabulka uchovávající informace o jednotlivých kontaktech, jako je např. titul, jméno, datum narození, IČ, rodné číslo, email, telefon, typ kontaktu (fyzická osoba, fyzická osoba – OSVČ, právnická osoba), atd. V aplikaci je tato tabulka reprezentována modelem `Contact.php`.

- **user**
  Tabulka uchovávající informace o jednotlivých uživatelích aplikace (pracovníci advokátní kanceláře). Obsahuje titul, jméno, přihlašovací jméno, heslo, telefon, cizí klíč na roli atd. V aplikaci je tato tabulka reprezentována modelem `User.php`.

- **file**
  Tabulka uchovávající informace o jednotlivých dokumentech ve spisech. Obsahuje název dokumentu, datum vytvoření, datum právní moci, cizí klíč na konkrétní spis, ve kterém se dokument nachází, atd. Dále může obsahovat i termín (např. do kdy může advokát podat odvolání), který si aplikace hlídá a upozorňuje na něj uživatele. V aplikaci je tato tabulka reprezentována modelem `File.php`.

- **meeting**
  Tabulka uchovávající informace o jednotlivých schůzkách. Obsahuje cizí klíče na kontakt a pracovníka, kterého se tato událost týká. Dále pak datum, poznámku, či případný dokument, ke kterému se schůzka pojí. V aplikaci je tato tabulka reprezentována modelem `Meeting.php`.

- **task**
  Tabulka uchovávající informace o jednotlivých úkolech pracovníků. Obsahuje cizí klíče na uživatele, který jej vytvořil a uživatele, který jej vykonává. Dále obsahuje popis, termín úkolu atd. V aplikaci je tato tabulka reprezentována modelem `Task.php`.

- **costs**
  Tabulka uchovávající informace o jednotlivých nákladech advokáta ve spise. Obsahuje název, datum, částku atd. V aplikaci je reprezentována modelem `Costs.php`.

- **reimbursement**
  Tabulka obsahující informace o jednotlivých platbách, které ve spise advokát přijal. Obsahuje datum, popis, částku atd. V aplikaci je reprezentována modelem `Reimbursements.php`.

- **Ostatní tabulky**

- address (`Address.php`)
- dokument_has_opponent (`DocumentOpponent.php`)
- contact_type (`ContactType.php`)
- country (`Country.php`)
- role (`Role.php`)
- delivery_type (`DeliveryType.php`)
- document_state (`DocumentState.php`)
- file_content (`FileContent.php`)
- file_direction (`FileDirection.php`)
- history (`History.php`)
- meeting_state (`Meeting.php`)
- operation (`Operation.php`)
- operation_type (`Operation.php`)
- reward (`Reward.php`)
- reward_type (`RewardType.php`)
- task_state (`TaskState.php`)

B Datový slovník

Atribut	Datový typ	PK	Nulový	Výchozí	Odkaz	Poznámka
id	int(11)	Ano	Ne			primární klíč adresy
street	varchar(255)	Ne	Ne			ulice
city	varchar(255)	Ne	Ne			město
postal_code	varchar(20)	Ne	Ne			poštovní směrovací číslo
country_id	int(11)	Ne	Ne		country -> id	cizí klíč - země
contact_id	int(11)	Ne	Ne		contact -> id	cizí klíč - kontakt, kterému adresa patří
delivery_type_id	int(11)	Ne	Ano	NULL	delivery_type -> id	cizí klíč - typ adresy

Tabulka 1: Datový slovník - tabulka address

Atribut	Datový typ	PK	Nulový	Výchozí	Odkaz	Poznámka
id	int(11)	Ano	Ne			primární klíč kontaktu
surname	varchar(45)	Ne	Ne			příjmení
title	varchar(45)	Ne	Ano	NULL		titul
deleted	tinyint(1)	Ne	Ano	0		příznak, že je kontakt smazán
lawyer	int(11)	Ne	Ano	NULL	contact -> id	cizí klíč - právní zástupce kontaktu
contact_type_id	int(11)	Ne	Ne		contact_type -> id	cizí klíč - typ osoby
data_box	varchar(10)	Ne	Ano	NULL		datová schránka
ic	varchar(10)	Ne	Ano	NULL		identifikační číslo
dic	varchar(12)	Ne	Ano	NULL		daňové identifikační číslo
first_name	varchar(45)	Ne	Ano	NULL		křestní jméno
birth_date	date	Ne	Ano	NULL		datum narození
identification_number	varchar(20)	Ne	Ano	NULL		rodné číslo
email	varchar(50)	Ne	Ano	NULL		email
phone	varchar(20)	Ne	Ano	NULL		telefon
client	int(11)	Ne	Ne			cizí klíč - typ vystupování
death	date	Ne	Ano	NULL		datum úmrtí/zániku

Tabulka 2: Datový slovník - tabulka contact

Atribut	Datový typ	PK	Nulový	Výchozí	Odkaz	Poznámka
id	int(11)	Ano	Ne			primární klíč typu osoby
name	varchar(45)	Ne	Ne			název typu

Tabulka 3: Datový slovník - tabulka contact_type

Atribut	Datový typ	PK	Nulový	Výchozí	Odkaz	Poznámka
id	int(11)	Ano	Ne			primární klíč nákladu
name	varchar(200)	Ne	Ne			popis
price	double	Ne	Ne			cena
date	date	Ne	Ne			datum vložení
document_id	int(11)	Ne	Ne		document -> id	cizí klíč - dokument, ke kterému náklad patří

Tabulka 4: Datový slovník - tabulka costs

Atribut	Datový typ	PK	Nulový	Výchozí	Odkaz	Poznámka
id	int(11)	Ano	Ne			primární klíč země
name	varchar(255)	Ne	Ne			název země

Tabulka 5: Datový slovník - tabulka country

Atribut	Datový typ	PK	Nulový	Výchozí	Odkaz	Poznámka
id	int(11)	Ano	Ne			primární klíč typu adresy
name	varchar(45)	Ne	Ne			název typu adresy
color	varchar(45)	Ne	Ne			barva typu adresy

Tabulka 6: Datový slovník - tabulka delivery_type

Atribut	Datový typ	PK	Nulový	Výchozí	Odkaz	Poznámka
id	int(11)	Ano	Ne			primární klíč spisu
number	int(11)	Ne	Ne			číslo spisu
year	int(11)	Ne	Ano	NULL		rok spisu
name	varchar(100)	Ne	Ano	NULL		název spisu
created	datetime	Ne	Ne	CURRENT_TIMESTAMP		datum a čas vytvoření
note	varchar(510)	Ne	Ano	NULL		poznámka
document_state_id	int(11)	Ne	Ano	NULL	document_state -> id	cizí klíč - stav spisu
owner	int(11)	Ne	Ne		user -> id	cizí klíč - majitel spisu
contact_id	int(11)	Ne	Ne		contact -> id	cizí klíč - klient ve spise
deleted	tinyint(1)	Ne	Ano	0		příznak, že je spis smazán

Tabulka 7: Datový slovník - tabulka document

Atribut	Datový typ	PK	Nulový	Výchozí	Odkaz	Poznámka
document_id	int(11)	Ne	Ne		document -> id	cizí klíč - spis
contact_id	int(11)	Ne	Ne		contact -> id	cizí klíč - protistrana

Tabulka 8: Datový slovník - tabulka document_has_opponent

Atribut	Datový typ	PK	Nulový	Výchozí	Odkaz	Poznámka
id	int(11)	Ano	Ne			primární klíč stavu spisu
name	varchar(45)	Ne	Ne			název stavu spisu
color	varchar(45)	Ne	Ne			barva stavu spisu

Tabulka 9: Datový slovník - tabulka document_state

Atribut	Datový typ	PK	Nulový	Výchozí	Odkaz	Poznámka
id	int(11)	Ano	Ne			primární klíč dokumentu
name	varchar(255)	Ne	Ne			název
sequence	int(11)	Ne	Ne			pořadové číslo
deleted	tinyint(1)	Ne	Ano	0		příznak, že je dokument smazán
document_id	int(11)	Ne	Ne		document -> id	cizí klíč - spis, kterému dokument patří
note	varchar(100)	Ne	Ano	NULL		poznámka
created	datetime	Ne	Ne	CURRENT_TIMESTAMP		datum a čas vytvoření
file_direction_id	int(11)	Ne	Ne		file_direction -> id	cizí klíč - směr dokumentu
pm	date	Ne	Ano	NULL		datum právní moci
received	date	Ne	Ne			datum přijetí
file_term_date	date	Ne	Ano	NULL		termín
file_term_note	varchar(100)	Ne	Ano	NULL		poznámka k termínu

Tabulka 10: Datový slovník - tabulka file

Atribut	Datový typ	PK	Nulový	Výchozí	Odkaz	Poznámka
id	int(11)	Ano	Ne			primární klíč přílohy
location	varchar(255)	Ne	Ne			umístění
created	datetime	Ne	Ne	CURRENT_TIMESTAMP		datum a čas vytvoření
deleted	tinyint(1)	Ne	Ne	0		příznak, že je příloha smazána
file_id	int(11)	Ne	Ne		file -> id	cizí klíč - dokument, kterému příloha patří
server_name	varchar(255)	Ne	Ne			název na serveru
user_name	varchar(255)	Ne	Ne			název dle uživatele
version	int(11)	Ne	Ne	1		verze přílohy

Tabulka 11: Datový slovník - tabulka file_content

Atribut	Datový typ	PK	Nulový	Výchozí	Odkaz	Poznámka
id	int(11)	Ano	Ne			primární klíč směru dokumentu
name	varchar(20)	Ne	Ne			název směru

Tabulka 12: Datový slovník - tabulka file_direction

Atribut	Datový typ	PK	Nulový	Výchozí	Odkaz	Poznámka
id	int(11)	Ano	Ne			primární klíč historie
action	varchar(255)	Ne	Ne			popis akce
date	datetime	Ne	Ne	CURRENT_TIMESTAMP		datum a čas, kdy akce proběhla
user_id	int(11)	Ne	Ne		user -> id	cizí klíč - uživatel, který akci vytvořil

Tabulka 13: Datový slovník - tabulka history

Atribut	Datový typ	PK	Nulový	Výchozí	Odkaz	Poznámka
id	int(11)	Ano	Ne			primární klíč schůzky
date	datetime	Ne	Ne			datum a čas schůzky
note	varchar(255)	Ne	Ano	NULL		poznámka
user_id	int(11)	Ne	Ne		user -> id	cizí klíč - uživatel, kterého se schůzka týká
meeting_state_id	int(11)	Ne	Ne		meeting_state -> id	cizí klíč - stav schůzky
contact_id	int(11)	Ne	Ne		contact -> id	cizí klíč - kontakt, kterého se schůzka týká
document_id	int(11)	Ne	Ano	NULL	document -> id	cizí klíč - dokument, kterého se schůzka týká

Tabulka 14: Datový slovník - tabulka meeting

Atribut	Datový typ	PK	Nulový	Výchozí	Odkaz	Poznámka
id	int(11)	Ano	Ne			primární klíč stavu schůzky
name	varchar(45)	Ne	Ne			název stavu
color	varchar(10)	Ne	Ne			barva stavu

Tabulka 15: Datový slovník - tabulka meeting_state

Atribut	Datový typ	PK	Nulový	Výchozí	Odkaz	Poznámka
id	int(11)	Ano	Ne			primární klíč advokátního úkonu
price	double	Ne	Ne			cena advokátního úkonu
reward_id	int(11)	Ne	Ne		reward -> id	cizí klíč - odměna, ke které úkon patří
operation_type_id	int(11)	Ne	Ne		operation_type -> id	cizí klíč - typ advokátního úkonu
date	datetime	Ne	Ne	CURRENT_TIMESTAMP		datum a čas vložení

Tabulka 16: Datový slovník - tabulka operation

Atribut	Datový typ	PK	Nulový	Výchozí	Odkaz	Poznámka
id	int(11)	Ano	Ne			primární klíč typu úkonu
name	varchar(255)	Ne	Ne			název typu
coeficient	double	Ne	Ne	1		koeficient, kterým se násobí cena odměny

Tabulka 17: Datový slovník - tabulka operation_type

Atribut	Datový typ	PK	Nulový	Výchozí	Odkaz	Poznámka
id	int(11)	Ano	Ne			primární klíč platby
name	varchar(100)	Ne	Ne			popis
date	datetime	Ne	Ne	CURRENT_TIMESTAMP		datum a čas vložení
price	double	Ne	Ne			cena platby
document_id	int(11)	Ne	Ne		document -> id	cizí klíč - spis, ke kterému platba patří
contact_id	int(11)	Ne	Ne		contact -> id	cizí klíč - kontakt, který platbu uhradil

Tabulka 18: Datový slovník - tabulka reimbursement

Atribut	Datový typ	PK	Nulový	Výchozí	Odkaz	Poznámka
id	int(11)	Ano	Ne			primární klíč odměny
reward_type_id	int(11)	Ne	Ne		reward_type -> id	cizí klíč - typ odměny
price	double	Ne	Ano	NULL		cena odměny
regular_expense	double	Ne	Ano	NULL		hotový výdaj
document_id	int(11)	Ne	Ne			cizí klíč - spis, ke kterému odměna patří

Tabulka 19: Datový slovník - tabulka reward

Atribut	Datový typ	PK	Nulový	Výchozí	Odkaz	Poznámka
id	int(11)	Ano	Ne			primární klíč typu odměny
name	varchar(45)	Ne	Ne			název typu

Tabulka 20: Datový slovník - tabulka reward_type

Atribut	Datový typ	PK	Nulový	Výchozí	Odkaz	Poznámka
id	int(11)	Ano	Ne			primární klíč role
name	varchar(45)	Ne	Ne			název role

Tabulka 21: Datový slovník - tabulka role

Atribut	Datový typ	PK	Nulový	Výchozí	Odkaz	Poznámka
id	int(11)	Ano	Ne			primární klíč úkolu
note	varchar(255)	Ne	Ne			poznámka
created	datetime	Ne	Ne	CURRENT_TIMESTAMP		datum a čas vytvoření
term	datetime	Ne	Ne			termín úkolu
created_by	int(11)	Ne	Ne		user -> id	cizí klíč - uživatel, který úkol vytvořil
performs	int(11)	Ne	Ne		user -> id	cizí klíč - uživatel, který úkol vykonává
document_id	int(11)	Ne	Ano	NULL	document -> id	cizí klíč - spis, kterého se úkol týká
task_state_id	int(11)	Ne	Ne		task_state -> id	cizí klíč - stav úkolu

Tabulka 22: Datový slovník - tabulka task

Atribut	Datový typ	PK	Nulový	Výchozí	Odkaz	Poznámka
id	int(11)	Ano	Ne			primární klíč stavu úkolu
name	varchar(45)	Ne	Ano		NULL	název stavu

Tabulka 23: Datový slovník - tabulka task_state

Atribut	Datový typ	PK	Nulový	Výchozí	Odkaz	Poznámka
id	int(11)	Ano	Ne			primární klíč uživatele
first_name	varchar(45)	Ne	Ne			křestní jméno
surname	varchar(45)	Ne	Ne			příjmení
username	varchar(45)	Ne	Ne			přezdívka
password	varchar(45)	Ne	Ne			heslo
deleted	tinyint(1)	Ne	Ne	0		příznak, že byl uživatel smazán
last_login	datetime	Ne	Ne	CURRENT_TIMESTAMP		datum a čas posledního přihlášení do IS
role_id	int(11)	Ne	Ne		role -> id	cizí klíč - role
title	varchar(45)	Ne	Ano	NULL		titul
phone	varchar(20)	Ne	Ano	NULL		telefon
email	varchar(50)	Ne	Ano	NULL		email
note	varchar(2048)	Ne	Ano	NULL		poznámka uživatele

Tabulka 24: Datový slovník - tabulka user

C Instalační příručka

Advokátní kancelář, která tento systém chce využívat, musí nejprve zvolit, zda chce využívat možnosti přístupu i přes internet. V případě, že ji stačí pouze přístup přes místní síť, odpadá ji tak část nastavování Apache serveru (konkrétně povolení přesměrování portu na směrovači). Popis instalace a nastavení programu EasyPHP na OS Windows lze najít v kapitole 5.

Po zprovoznění webového serveru je potřeba celou složku /advokatis zkopírovat z přiloženého CD do složky /projects v projektovém adresáři EasyPHP serveru (typicky adresář /data/localweb). Výsledná struktura by tedy měla vypadat např. takto /data/localweb/projects/advokatis. Zprovoznění Zend frameworku již není potřeba, protože je celý obsažen v aplikaci. Je také nutné správně nastavit konfigurační soubory `httpd.conf` a `php.ini`. Jejich nastavení je popsáno v kapitole 5.

Dále je nutné vytvořit databázi. SQL skript pro její vytvoření se nachází na přiloženém CD (/database/createDB.sql). Tento skript vytvoří veškeré tabulky databáze a vloží do ní prvního uživatele (přihlašovací jméno: admin, heslo: 1234). Pomocí tohoto uživatele poté advokát, nebo správce systému vytvoří uživatele, kteří budou s aplikací pracovat.

Aby aplikace komunikovala s již vytvořenou databází, je potřeba informace o databázi vložit do konfiguračního souboru `config.ini`, který se nachází ve složce /application/configs. Ukázku připojení se k databázi pomocí konfiguračního souboru lze nalézt ve výpise č. 1.

Poté, co jsou zadány správné přihlašovací údaje k databázi, lze se do aplikace přihlásit zadáním adresy http://127.0.0.1/projects/advokatis/public. V případě, že se uživatel chce přihlásit z jiného počítače, který se nachází v místní síti, je potřeba do prohlížeče zadat místní IP adresu serveru, na kterém aplikace ADVOKATIS běží.

D Uživatelská příručka

D.1 První spuštění aplikace

Při prvotním spuštění aplikace je potřeba se přihlásit přes již vytvořený administrátorský účet.

Přihlašovací jméno:	**admin**
Heslo:	**1234**

D.2 Hlavní rozdělení aplikace

- V pravé horní části jsou notifikační ikony a informace o uživateli.
- V levé části okna se nachází univerzální vyhledávací formulář a navigace.
- Ve střední části okna se zobrazuje obsah konkrétních stránek.

V pravé horní části okna jsou tři ikony. První zobrazuje nejbližší termíny jednotlivých dokumentů, druhá ukazuje nejbližší neukončené úkoly momentálně přihlášeného uživatele. Pomocí třetí ikony lze zobrazit informace o přihlášeném uživateli, nebo se odhlásit z aplikace. Pokud v aplikaci existuje nějaký neproběhlý termín nebo neukončený úkol uživatele, jsou tyto ikony zbarveny červeně.

V levé horní části okna se nachází univerzální vyhledávací formulář. Tento formulář slouží pro vyhledávání ve spisech a kontaktech. Obsahuje jedno textové pole pro zadání hodnoty a dvě tlačítka. První pro vyhledávání ve spisech, druhé pro vyhledávání v kontaktech. Ve spisech lze vyhledávat pomocí čísla a roku konkrétního spisu ("1/15"), podle seznamu jednotlivých spisů ("3/14, 1/15, 8/15"), nebo podle konkrétního roku ("/15"), kdy jsou zobrazeny jen spisy s tímto rokem. Při vyhledávání v kontaktech lze zadat buď datum narození, rodné číslo, IČ, jméno, nebo jméno a příjmení.

Pro zobrazení mnoha záznamů je v aplikaci ADVOKATIS využito dynamických tabulek, které umožňují záznamy různě řadit kliknutím na název sloupce. Lze použít i řazení pomocí dvou a více sloupců, kdy je pro jejich výběr potřeba podržet tlačítko SHIFT a kliknout na název sloupce.

Některé sloupce mohou i s konkrétním záznamem obsahovat ikonu dvou malých lomených závorek (>>), která znamená, že po jejím stisknutí budete přesměrováni na konkrétní záznam z tohoto sloupce. Takovéto odkazy jsou využívány napříč celou aplikací ADVOKATIS a nejen v tabulkách.

D.3 Navigace

- **Úvod**
  Na této stránce se nachází Seznam nejnovějších spisů a Poznámkový blok uživatele. Tento blok slouží pro vytvoření rychlé poznámky. Každý uživatel má svou vlastní poznámku. Ihned, co uživatel dopíše poznámku a opustí její textové pole, je tato hodnota trvale uložena.

- **Spisy**
  Tato sekce má na starost práci s veškerými spisy.

  - **Seznam spisů:** Na této stránce lze zobrazit seznam všech nesmazaných spisů. Obsahuje také pole pro rychlé vyhledávání v těchto spisech.
  - **Nový spis:** Stránka pro vytvoření nového spisu. Je potřeba zadat povinné informace o spise, např. číslo, rok atd. Nelze vytvořit dva spisy se stejným číslem a rokem. Pro přiřazení jednotlivých kontaktů a uživatelů k vytvářenému spisu je nutné, aby tyto osoby již v aplikaci existovaly. Po úspěšném vytvoření je uživatel přesměrován na detail spisu.
  - **Detail spisu (nenachází se v navigaci):** Pravděpodobně nejdůležitější stránka aplikace, která obsahuje kompletní informace o vybraném spise. Pokud je číslo spisu zbarveno červeně, znamená to, že se jedná o smazaný spis a jsou v něm znemožněny jakékoliv úpravy. Detail spisu je rozdělen na následující sekce:
    * **Základní informace:** Obsahuje parametry spisu a tlačítka pro upravení a smazání spisu.
    * **Účastníci:** Obsahuje klienta a protistrany vystupující v tomto spise. Také obsahuje tlačítka pro vybrání jiného klienta a přidání protistrany do spisu. Každý záznam v tabulce protistrany obsahuje v posledním sloupci ikonu křížku, pomocí kterého se dá vybraná protistrana odebrat z tohoto spisu.
    * **Dokumenty:** Obsahuje seznam všech dokumentů týkajících se tohoto spisu a tlačítko „Přidat“ pro vytvoření nového dokumentu. Pokud je v názvu karty Dokumenty červená hvězda, znamená to, že ve spise již nějaké dokumenty existují. Každý dokument má na svém řádku ikonu symbolizující lupu pro zobrazení detailu dokumentu, ikonu symbolizující tužku pro upravení a ikonu kříže pro smazání. Jednotlivé dokumenty umožňují připojit přílohy, které lze verzovat. Tyto přílohy lze stáhnout nebo smazat v detailu dokumentu. V případě, že je text ve sloupci Termín zbarven červeně, znamená to, že datum tohoto termínu již proběhlo.
    * **Advokátní tarif:** Obsahuje informace o typu odměny pro tento spis. V případě, že spis ještě nemá odměnu nastavenou, zobrazí se tlačítko pro vytvoření odměny. V případě, že spis obsahuje Mimosmluvní odměnu, zobrazí se také v této kartě seznam jednotlivých advokátních úkonů a tlačítka pro jejich vytváření, úpravu a smazání. Každý typ odměny má tlačítko pro úpravu.

    - **Platby:** V této kartě jsou evidovány veškeré příchozí platby týkající se spisu. Pokud je v názvu karty Platby červená hvězda, znamená to, že ve spise již nějaké platby existují. Dále jsou zde obsažena tlačítka pro vytváření, úpravu a smazání platby.
    - **Náklady:** V této kartě jsou evidovány veškeré náklady (výdaje advokáta) týkající se spisu. Pokud je v názvu karty Náklady červená hvězda, znamená to, že ve spise již nějaké náklady existují. Dále jsou zde obsažena tlačítka pro vytváření, úpravu a smazání nákladu.
    - **Úkoly:** Tato karta obsahuje veškeré úkoly týkající se spisu. Úkoly jsou dvojího typu: aktivní a ukončené. Pokud je úkol ukončen, celý řádek tohoto úkolu je podbarven žlutě. V případě, že je text ve sloupci Termín zbarven červeně, znamená to, že datum tohoto termínu již proběhlo. Úkoly v této kartě lze vytvářet, ukončit, upravit a smazat.

- **Kontakty:** Tato sekce má na starost práci s veškerými kontakty (klienti, protistrany a právní zástupci).

  - **Seznam kontaktů:** Na této stránce lze zobrazit seznam všech nesmazaných kontaktů. Obsahuje také pole pro rychlé vyhledávání v těchto kontaktech.
  - **Nový kontakt:** Stránka pro vytvoření nového kontaktu. Zprvu je potřeba vybrat typ osoby (fyzická, fyzická-OSVČ, právnická) a poté je zobrazen formulář s ostatními údaji pro tento vybraný typ. V případě, že je vybrán typ osoby fyzická-OSVČ, nebo právnická, je zobrazeno tlačítko pro načtení podrobných informací o kontaktu z veřejných rejstříků ARES a PVS. Pro tento způsob načtení dat je potřeba vyplnit IČ kontaktu. Aplikace při vytváření kontroluje, zda-li již v systému neexistuje kontakt se stejným identifikátorem (RČ, nebo IČ). Po úspěšném vytvoření je uživatel přesměrován na detail kontaktu.
  - **Detail kontaktu (nenachází se v navigaci):** Obsahuje kompletní informace o vybraném kontaktu. Pokud je název kontaktu zbarven červeně, znamená to, že se jedná o smazaný kontakt a jsou v něm znemožněny jakékoliv úpravy. Detail kontaktu je rozdělen na následující sekce:
    - **Základní informace:** Obsahuje parametry spisu a tlačítka pro upravení a smazání kontaktu.
    - **Adresy:** Obsahuje seznam adres připojených ke kontaktu. Adresy lze přidávat, upravovat a mazat. Pokud je v názvu karty Adresy červená hvězda, znamená to, že u kontaktu již nějaké adresy existují.
    - **Spisy:** Obsahuje seznam spisů, ke kterým je kontakt připojen. Pokud je v názvu karty Spisy červená hvězda, znamená to, že kontakt je připojen alespoň k jednomu spisu.
    - **Schůzky:** Tato karta obsahuje veškeré schůzky týkající se kontaktu. Schůzky mohou mít tři stavy: neproběhlá, proběhlá a zrušená. Pokud je stav schůzky proběhlá, nebo zrušená, celý řádek této schůzky je podbarven žlutě. V případě, že je text ve sloupci Datum zbarven červeně, znamená to, že datum

této schůzky již proběhlo. Schůzky lze v této kartě vytvářet, upravit a mazat.

- **Úkoly:** Tato část obstarává veškerou práci s úkoly.
  - **Vaše úkoly:** Obsahuje seznam všech úkolů, patřících přihlášenému uživateli. Úkoly zde lze řadit, vyhledávat, ukončit, upravit a smazat. Červené datum ve sloupci Termín značí, že termín tohoto úkolu už vypršel. Pokud je celý řádek úkolu podbarven žlutě, znamená to, že úkol je již ukončený.
  - **Úkoly všech:** Obsahuje seznam všech úkolů. Úkoly zde lze řadit, vyhledávat, ukončit, upravit a smazat. Červené datum ve sloupci termín značí, že termín tohoto úkolu už vypršel. Pokud je celý řádek úkolu podbarven žlutě, znamená to, že úkol je již ukončený.
  - **Nový úkol:** Na této stránce lze vytvořit nový úkol. Pro jeho úspěšné vytvoření je potřeba vyplnit termín novější, než je datum aktuální (nelze tedy vytvořit úkol do minulosti).
- **Schůzky:** Tato část obstarává veškerou práci se schůzkami.
  - **Vaše schůzky:** Obsahuje seznam všech schůzek patřících přihlášenému uživateli. Schůzky zde lze řadit, vyhledávat, upravit a smazat. Červené datum ve sloupci Datum značí, že termín této schůzky už vypršel. Pokud je celý řádek schůzky podbarven žlutě, znamená to, že schůzka již proběhla, nebo byla zrušena.
  - **Všechny schůzky:** Obsahuje seznam všech schůzek. Schůzky zde lze řadit, vyhledávat, upravit a smazat. Červené datum ve sloupci Datum značí, že termín této schůzky už vypršel. Pokud je celý řádek schůzky podbarven žlutě, znamená to, že schůzka již proběhla, nebo byla zrušena.
  - **Nová shůzka:** Na této stránce lze vytvořit novou schůzku. Pro její úspěšné vytvoření je potřeba, stejně jako u úkolů, vyplnit datum novější, než je datum aktuální.
- **Termíny:** Obsahuje seznam všech nenastalých termínů k jednotlivým dokumentům ve spisech. Ve sloupci Průběh zobrazuje počet zbývajících dnů, než tento termín nastane. Pokud je položka v tomto sloupci zobrazena zeleně, znamená to, že termín nastane za více než 9 dní. Pokud je položka zbarvena žlutě, znamená to, že termín nastane za 5 až 9 dní. Pokud je položka zbarvena červeně, znamená to, že termín nastane za méně než 5 dní.
- **Kalendář:** Na této stránce je zobrazen kalendář veškerých událostí (termíny, úkoly a schůzky). Po výběru konkrétního data se načtou události pro daný den. Lze zde také vytvářet nové úkoly a schůzky. Pokud má uživatel vybráno datum, které ještě neproběhlo, je toto datum automaticko vyplněno do pole Datum, nebo Termín při vytváření nové schůzky či úkolu, v opačném případě je kolonka prázdná a uživatel musí datum/termín/ vyplnit ručně.

- **Nastavení:** V této části lze spravovat veškeré číselníky, advokátní tarif, uživatele a zobrazit historii.
  - **Číselníky:** Lze zde vytvářet nové stavy spisů a typy adres. Taktéž je možné tyto stavy a typy smazat, nebo upravit. Dále se zde dá upravit barva jednotlivých stavů schůzek.
  - **Advokátní tarif:** Na této stránce lze vytvářet, upravovat a mazat úkony, pojící se k mimosmluvní odměně advokáta.
  - **Uživatelé:** Zobrazuje seznam všech aktivních uživatelů v systému. Taktéž je zde možné vytvářet nové uživatele.
  - **Detail uživatele (nenachází se v navigaci):** V této kartě jsou zobrazeny veškeré informace o uživateli. Pokud je zobrazen detail aktuálně přihlášeného uživatele, má tento uživatel možnost upravit své osobní informace, nebo si změnit heslo. V případě, že je přihlášen Advokát, nebo Administrátor, má možnost jakéhokoliv uživatele upravit, smazat, nebo mu jen změnit heslo. V případě akce smazání nelze odstranit sama sebe.
  - **Historie:** Zobrazuje seznam a popis veškerých akcí, které uživatelé v systému provedli.

E Obsah přiloženého CD

Přiložené CD obsahuje následující složky:

- **/advokatis**
  Složka s aplikací ADVOKATIS.
- **/database**
  Složka s sql skriptem pro vytvoření databáze.
- **/docs**
  Složka s dokumentací aplikace ADVOKATIS. Tato dokumentace byla vygenerována pomocí programu Apigen [29].